

NASA Technical Memorandum 105186

**The Navy/NASA Engine Program  
(NNEP89)—A User's Manual**

**Robert M. Plencner and Christopher A. Snyder**  
*Lewis Research Center*  
*Cleveland, Ohio*

(NASA-TM-105186) THE NAVY/NASA ENGINE  
PROGRAM (NNEP89): A USER'S MANUAL (NASA)  
118 p CSCL 21E

N91-30141

Unclas  
G3/07 0033605

**August 1991**

**NASA**

## Table of Contents

SUMMARY.....	1
1.0 INTRODUCTION.....	3
2.0 GENERAL CODE DESCRIPTION.....	5
3.0 COMPONENT PERFORMANCE MAPS.....	7
3.1 COMPRESSOR PERFORMANCE MAPS.....	8
3.2 TURBINE PERFORMANCE MAPS.....	8
3.3 VARIABLE SCHEDULE PERFORMANCE MAPS.....	12
4.0 CODE INPUT DESCRIPTIONS.....	13
5.0 NNEP89 ENGINE MODEL DEVELOPMENT.....	15
5.1 CONFIGURING AN ENGINE.....	15
5.1.1 Components and Flow Stations.....	15
5.1.2 Secondary Flow Streams.....	18
5.2 MULTIMODE ENGINE CONFIGURATIONS.....	19
5.3 OFF-DESIGN OPERATION.....	19
5.4 REDESIGNING COMPONENTS DURING OFF-DESIGN OPERATION.....	21
5.5 CONFIGURATION LIMITATIONS.....	22
6.0 ENGINE COMPONENTS.....	23
6.1 INLET.....	23
6.2 DUCT or BURNER.....	23
6.3 GAS GENERATOR.....	24
6.4 WATER INJECTOR.....	24
6.5 COMPRESSOR or FAN.....	25
6.6 TURBINE.....	25
6.7 HEAT EXCHANGER.....	25
6.8 SPLITTER.....	26
6.9 MIXER or EJECTOR.....	26
6.10 NOZZLE.....	26
6.11 LOAD or PROPELLER.....	27
6.12 SHAFTS.....	27
7.0 PROGRAM CONTROL COMPONENTS.....	29
7.1 VARIABLE CONTROL.....	29
7.2 VARIABLE OPTIMIZATION.....	29
7.3 VARIABLE LIMITER.....	29
7.4 VARIABLE SCHEDULING.....	29
7.5 CONDITIONAL CONTROL.....	30
8.0 CHEMICAL EQUILIBRIUM.....	31
9.0 SIMPLIFIED INSTALLATION CALCULATIONS.....	33
9.1 INLET DRAGS.....	33
9.2 NOZZLE DRAG.....	35
10.0 INPUT/OUTPUT DESCRIPTIONS.....	37
10.1 GLOBAL &D NAMELIST INPUTS.....	37
10.1.1 Input/Output.....	37
10.1.2 Execution Control.....	38
10.1.3 Optimization.....	39
10.1.4 Turbine Cooling.....	39
10.1.5 Thermodynamic Properties (Including CEC).....	40
10.1.6 Installation.....	40
10.2 COMPONENT KONFIG AND SPEC INPUTS AND DATOUT OUTPUTS.....	41
10.2.1 Inlet.....	42
10.2.2 Duct or Burner.....	44
10.2.3 Gas Generator.....	47
10.2.4 Water Injector.....	49

10.2.5 Fan or Compressor .....	50
10.2.6 Turbine .....	52
10.2.7 Heat Exchanger .....	55
10.2.8 Flow Splitter .....	57
10.2.9 Mixer or Ejector .....	58
10.2.10 Nozzle .....	60
10.2.11 Load or Propeller .....	64
10.2.12 Shaft .....	66
10.2.13 Variable Control .....	67
10.2.13.1 Variable Target Values .....	70
10.2.13.2 Marching .....	70
10.2.14 Variable Optimization .....	72
10.2.15 Variable Limit .....	74
10.2.16 Variable Scheduling .....	75
10.2.17 Conditional Control .....	77
10.3 TABLE DATA INPUTS FORMAT .....	80
10.4 CHEMICAL EQUILIBRIUM INPUT INSTRUCTIONS .....	81
10.4.1 Using The CEC Debugging Parameters .....	82
10.4.2 Example Case Using CEC Routines .....	83
10.4.3 Using The IDBUG Option Without Using CEC Option .....	85
11.0 NNEP89 FLOW-CHART .....	87
12.0 SAMPLE INPUT AND OUTPUT .....	93
12.1 SAMPLE 1 - TWO-SPOOL MIXED-FLOW TURBOFAN .....	94
12.2 SAMPLE 2 - SEPARATE FLOW TURBOFAN USING CEC .....	96
12.3 SAMPLE 3 - ONE-SPOOL MIXED-FLOW TURBOFAN .....	97
12.4 SAMPLE 4 - AIR-TURBO RAMJET, USING CEC .....	99
12.5 SAMPLE 5 - TANDEM FAN WITH MULTIMODES .....	101
12.6 EXAMPLE OUTPUT FOR SAMPLE CASE 3 .....	106
12.7 EXAMPLE COMPRESSOR MAP TABLES .....	112
12.8 EXAMPLE TURBINE MAP TABLES .....	116
ACKNOWLEDGEMENT .....	119
REFERENCES .....	119

## SYMBOLS

A	Area, in <sup>2</sup> or ft <sup>2</sup>
C <sub>D</sub>	Drag coefficient
C-D	Converging-Diverging
D	Propeller diameter, ft
JM1	Main upstream flow station
JM2	Secondary upstream flow station
JP4	"Typical" hydrocarbon or kerosene aircraft fuel
JP1	Main downstream flow station
JP2	Secondary downstream flow station
M	Momentum
N	Rotational speed, revolutions per minute
n	Rotational speed, radians per second
NO <sub>x</sub>	Nitrous oxides
P	Total pressure, psia
PR	Pressure ratio
R	Rankine, Ideal gas constant, or arbitrary lines on compressor performance maps
SHP	Shaft power, hp
T	Total temperature, degrees Rankine
V	Velocity, ft/s
W	Weight flow, lb/s
$w\sqrt{T}/P$	Referred weight flow, lb/s
$w\sqrt{\theta}/\delta$	Corrected weight flow, lb/s
$\delta$	Corrected pressure, $\frac{\text{Actual pressure in psia}}{14.696}$
$\Delta$	Delta or change in

$f$	Fraction
$\dot{m}$	mass flow, lb/s
$\eta$	Efficiency
$\theta$	Corrected temperature, $\frac{\text{Actual total temperature in Rankine}}{518.67}$
$\rho$	Mass density, lb/ft <sup>3</sup>
$\pi$	Pi, 3.14159
$q$	Dynamic pressure, lb/ft <sup>2</sup>

#### Subscripts

actual	Actual or desired value
BLD	Inlet bleed
BOAT	Nozzle boattail
C	Capture
corr	Corrected
EXIT	Conditions at the exit of a component or engine
JM1	Main upstream flow station
JM2	Secondary upstream flow station
JP1	Main downstream flow station
JP2	Secondary downstream flow station
LIP	Inlet lip
MAXEN	Engine maximum cross-sectional area
map	value from component performance map
R	Ratio
SPL	Inlet spillage
V	Velocity
0	Free stream

# The Navy/NASA Engine Program (NNEP89) - A User's Manual

By Robert M. Plencner and Christopher A. Snyder

National Aeronautics and Space Administration  
Lewis Research Center  
Cleveland, Ohio 44136

## SUMMARY

An engine simulation computer code called NNEP89 has been written to perform one dimensional steady state thermodynamic analysis of turbine engine cycles. By using a very flexible method of input, a set of standard components are connected at execution time to simulate almost any turbine engine configuration that the user could contemplate. The code has been used to simulate a wide range of engine cycles from turboshafts and turboprops to air-turbo-rockets and supersonic cruise variable cycle engines. Off-design performance is calculated through the use of component performance maps. A chemical equilibrium model is incorporated to adequately predict chemical dissociation as well as model virtually any fuel. NNEP89 is written in standard FORTRAN77 with clear structured programming and extensive internal documentation. The standard FORTRAN77 programming allows it to be installed onto most mainframe computers and workstations without modification.

The NNEP89 code has been derived from the Navy/NASA Engine Program (NNEP). NNEP89 provides many improvements and enhancements to the original NNEP code and incorporates features which make it easier to use for the novice user. NNEP89 has been written to execute the old NNEP input files without changes to the program or the input files.

This report serves as a comprehensive user's guide for the NNEP89 code. It incorporates a general description of the code, comprehensive input description, program flow charts and sample input and output cases.

## 1.0 INTRODUCTION

In 1975 the NASA Lewis Research Center in conjunction with the Naval Air Development Center developed an engine cycle simulation computer code called the Navy/NASA Engine Program, NNEP (ref. 1). The NNEP code expanded greatly upon the capabilities of an existing Navy code, NEPCOMP, (ref. 2) by introducing multiple modes of operation to simulate variable cycle engines, "stacked" component maps for variable geometry components, and optimization capability. The program could therefore simulate the steady-state design and off-design performance of almost any turbine engine that the user could contemplate.

Projected increases in engine material capabilities and recent studies of air breathing engines for very high speed flight have created interest in engine cycles and engine conditions that NNEP could not handle adequately. First, very high temperatures are reached in many of these engine cycles. At these high temperatures, chemical dissociation of some of the engine gas streams can occur, which NNEP can not model. Therefore, the program was enhanced by adding a chemical dissociation model to NNEP. This same model allows NNEP to model cycles using any fuels including cryogenic fuels and slurries. Second, new component models were needed for certain innovative cycles. These new models enable the program to simulate, among other things, air-turbo rockets, ejector mixers, and rockets. A new version of NNEP was written to incorporate these features. This new version was called NNEPEQ, for NNEP with Equilibrium effects (ref. 3&4).

After the addition of chemical equilibrium, several other changes were desired, but the original structure of the program made such enhancements difficult. Therefore the program was rewritten using FORTRAN77 standards, with clearer, more structured programming, and with extensive internal documentation within the FORTRAN coding. This version of the program is called NNEP89 and includes many capabilities not in the original program. These capabilities include the use of propeller performance maps (ref. 5); the plotting of compressor, turbine, and propeller component maps and engine operating points on these maps (ref. 6); more options for improved nozzle performance modeling (ref. 7); a new component which will conditionally activate and deactivate a control component; programming to automatically set up certain program control parameters (ref. 8), and an option to calculate the amount of turbine cooling bleed flow requirements (ref. 9). These improvements make the program easier to use and add further enhancements. NNEP89 will still execute the old NNEP or NNEPEQ input datasets without changes to the program or the input datasets. This version of the code was also written to greatly ease its installation onto most computers.

This document provides a general description of the NNEP89 code and all the information required for the user to model any desired engine configuration. It contains sections that discuss the use of performance maps to predict component performance, the method of configuring an arbitrary engine configuration, a brief description of all the types of components that are available in the code, a description of the chemical equilibrium option, an explanation of installation loss calculations and a series of flow charts that describe the overall logic of the code. The majority of this document is dedicated to providing a detailed description and definition of all the program inputs. The end of this document contains many samples to aid the user in developing his own NNEP89 engine model.

Work is currently in progress to develop a NNEP89 programmer's manual. When completed, this document will provide details on the code theory, structure, logic, and operations.

## 2.0 GENERAL CODE DESCRIPTION

The NNEP89 engine simulation computer code performs one dimensional, steady state, thermodynamic analysis of turbine engine cycles. By using a very flexible method of input, a set of standard components are connected at execution time to simulate almost any turbine engine configuration that the user could contemplate. Off-design performance is calculated through the use of component performance maps. The compressor and turbine performance maps are scaled by the code to match the design point pressure ratio, corrected weight flow and efficiency of the engine being modeled. Engines that change configurations over various portions of their flight regimes are modeled using a feature of the code that allows the user to define multiple configurations of the same engine. These engines with multiple configurations are called multimode engines. The default thermodynamic routine used in the code is preset for mixtures of air and JP4 fuel. A chemical equilibrium model is incorporated as an option to adequately predict thermodynamic properties when chemical dissociation occurs as well as when using virtually any fuel.

In general, two separate input files are required to run the code. The first contains inputs which tell the code what components will be used and how those components are configured to form a specific engine model. Detailed inputs for each of these components describe the desired component model. Also included in this input file are global inputs which control program input/output, execution, optimization, turbine cooling, thermodynamic property calculations, and installation effects calculations. The second input file contains all performance map tables which are generally used to model off design performance of components such as compressors and turbines. If only design point calculations are being computed the map tables file is usually not required.

Although NNEP89 only does a thermodynamic analysis, it has been coupled to a number of other codes to extend this capability. The capability to estimate engine weights was added to the program (ref. 10). Presently, this is accomplished by using additional program libraries to perform the weight analysis. The weight program individually weighs each component, using user input for many design parameters. There is an option for graphical output of the flow path and interactive weight determination.

Simplified installation effects calculation are built into the program to give a preliminary estimate of inlet drag and nozzle drag. As an alternative to these internal installation calculations there is an additional program library which performs more detailed installation effects calculations (ref. 11). This additional program library can utilize much more sophisticated inlet and nozzle performance maps, to more accurately estimate installation losses and their effect on overall engine performance. The installation package can also perform much more detailed inlet and nozzle weights if sufficient inlet and nozzle system information is known.



### 3.0 COMPONENT PERFORMANCE MAPS

One of NNEP89's greatest strengths is its capacity to interpolate user-supplied data tables to estimate component performance. These data tables consist of a dependent variable which is a function of up to three independent variables (dependent variable =  $F(x,y,z)$ ). Consider the table data as three dimensional, composed of a series of planes with each plane assigned a value call Z. Then, on each Z plane, the dependent variable (ordinate axis), is a two dimensional function of X (abscissa axis) and Y (see Figure 1). These three dimensional tables are often referred to as maps. The data format for inputting all tables is given in Section 10.3.

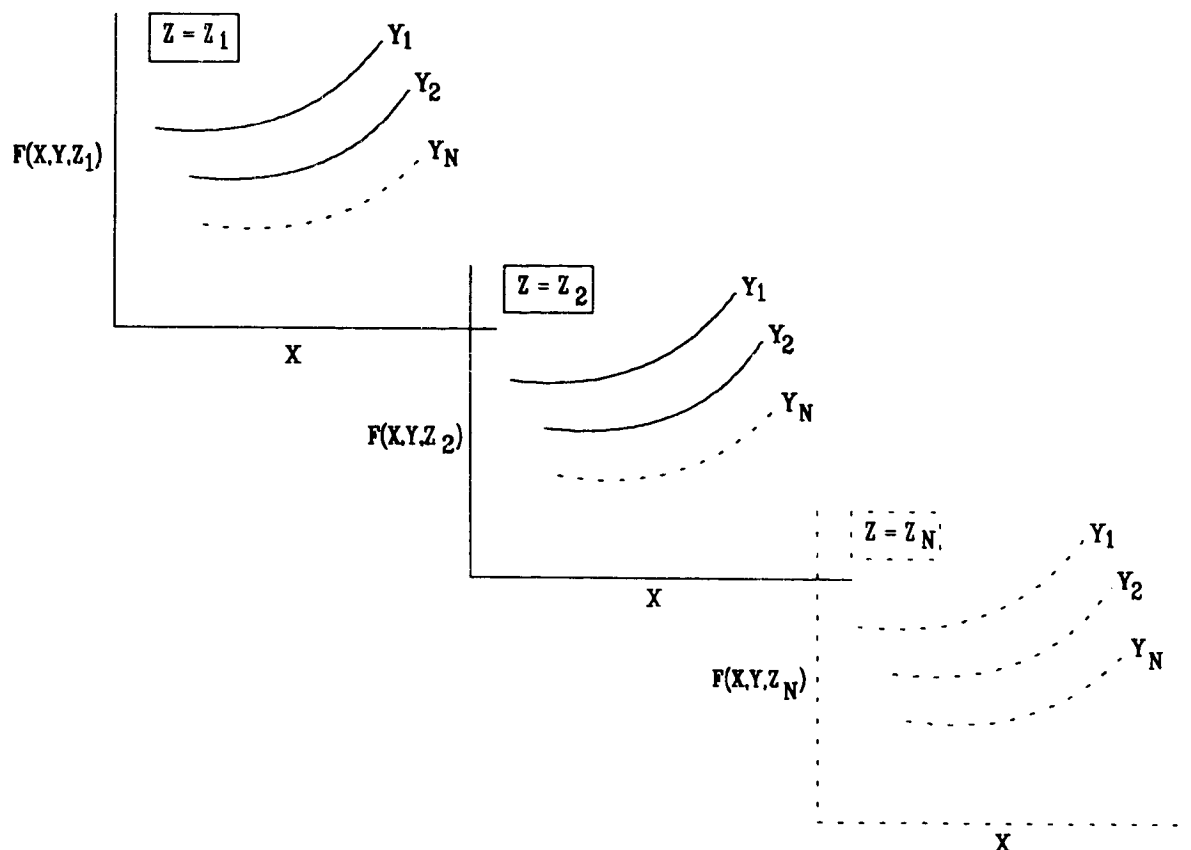


Figure 1. Three Dimensional Map Representation

Most of the engine components have inputs which may be specified in terms of a user input tabular function of one to three independent variables. The specific independent variables used are predetermined by the program. These functions are discussed in the descriptions of each of the components in Section 6. The user can also set up a variable schedule, which allows the user to set up a data table which sets any input variable to be a function of up to three other component inputs, component outputs, or engine flow station properties that the user specifies through the variable schedule inputs.

All the map tables required for a given engine model must be contained in a separate map file. This entire map file is read in by the program at the beginning of the execution of a set of cases and the entire file is stored in an array. This array is later interpolated by the code using a cubic spline interpolation technique to get the values of the dependent variables corresponding to the current values of the independent variables. If an independent variable is outside the range of the tabular data, a linear extrapolation will be performed using the slope of the spline at the last point in the table.

The map file may contain additional map tables that are not used by a specific engine model. The only restrictions are that the total number of map tables in the file is no more than thirty and no more than 20000 array locations are required to store all the map tables. The number of array locations required to store the data is roughly four times higher than the actual number of data points being stored. Therefore, when using very large map files for the first time, the user is cautioned to check the output for error messages to ensure all tables have been stored properly.

### 3.1 COMPRESSOR PERFORMANCE MAPS

A typical compressor map is shown in Figure 2. The third dimensional argument not shown in the figure is stator angle. Thus there are a series of plots similar to the one shown in Figure 2 each with a different value of stator angle. However, note that for a given corrected rotational speed and stator angle, there is a family of values for compressor pressure ratio, corrected mass flow, and adiabatic efficiency. The program therefore uses arbitrary lines on the maps called R lines, as shown in Figure 3. These R lines are drawn by the user following the general shape of the surge line. R lines must not cross or intersect each other. Each R line is assigned an arbitrary value with increasing values from the surge line. The surge line is generally (but not necessarily) defined as an R value of one. The three map properties (pressure ratio, corrected flow, and efficiency) can then be defined in tabular form as a function of corrected speed, R value, and stator angle. At the engine design point, the user specifies the design point on the map by specifying R value, corrected rotational speed, and stator angle. The user also inputs the desired compressor pressure ratio and efficiency. This defines unique compressor map values for pressure ratio, corrected mass flow, and efficiency. For the design point calculation, the program will use the compressor performance input by the user and the design-point specified on the component maps and calculate map scale factors. The equations for the map scale factors are given in Section 10.2.5. The map scale factors will be used by the program for off-design points to convert the compressor corrected mass flow, pressure ratio, and efficiency interpolated from the performance maps into the actual performance properties to be used by the program.

There are separate programs available for generating compressor and fan performance maps (ref. 12 & 13). The programs are easy to use and can generate compressor and fan performance maps with operating characteristics tailored for the user's application. These generated performance maps are already in the NNEP89 format.

### 3.2 TURBINE PERFORMANCE MAPS

Typical turbine performance maps are shown in Figure 4. Although the use and scaling of turbine performance maps are similar to compressor performance maps, there are some notable differences. At the design point the user specifies the design point on the turbine performance maps (by specifying the turbine map pressure ratio and its corrected rotational speed) and the desired turbine efficiency. At the design point, the program will determine the turbine pressure ratio required to balance the work requirements on that shaft. Using this

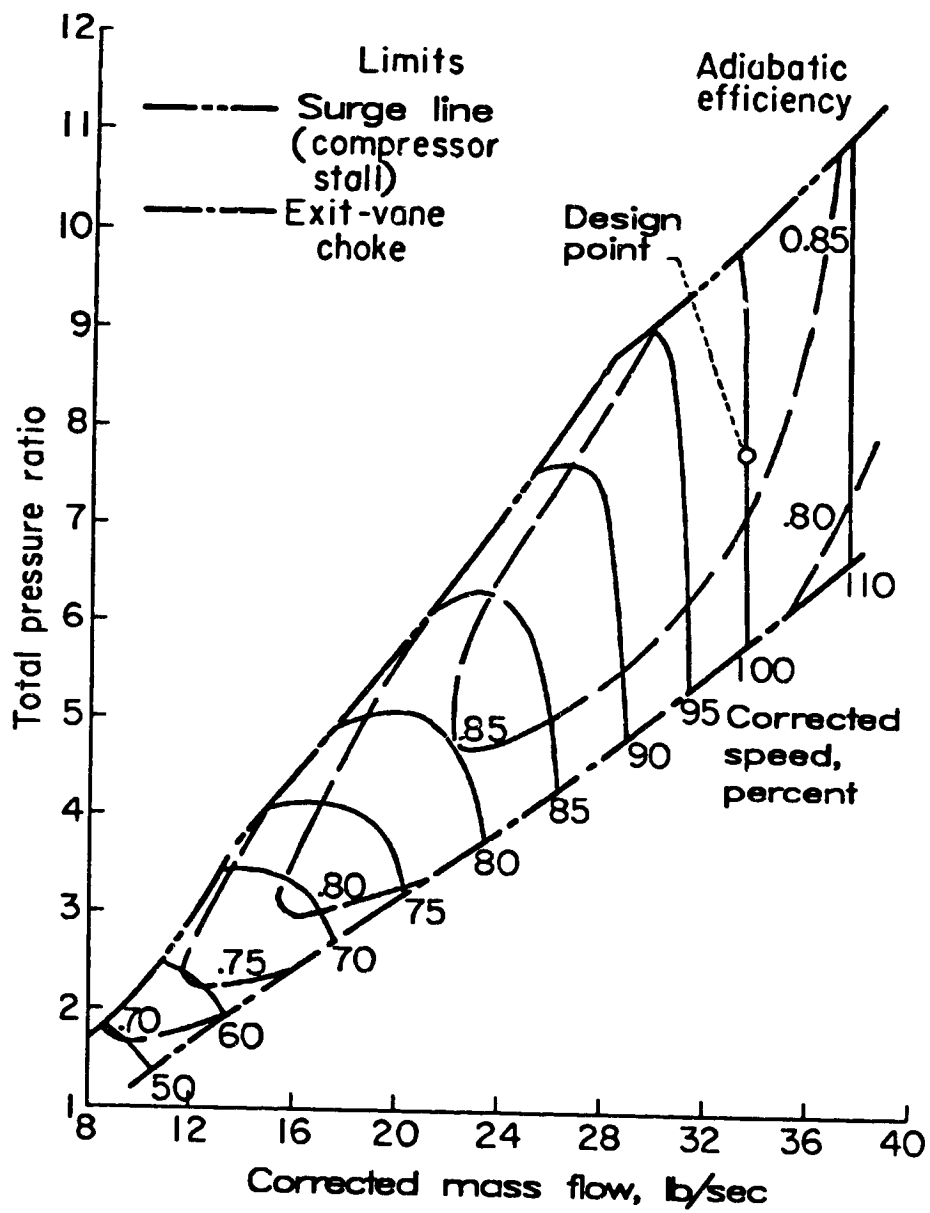


Figure 2. Typical Compressor Performance Map

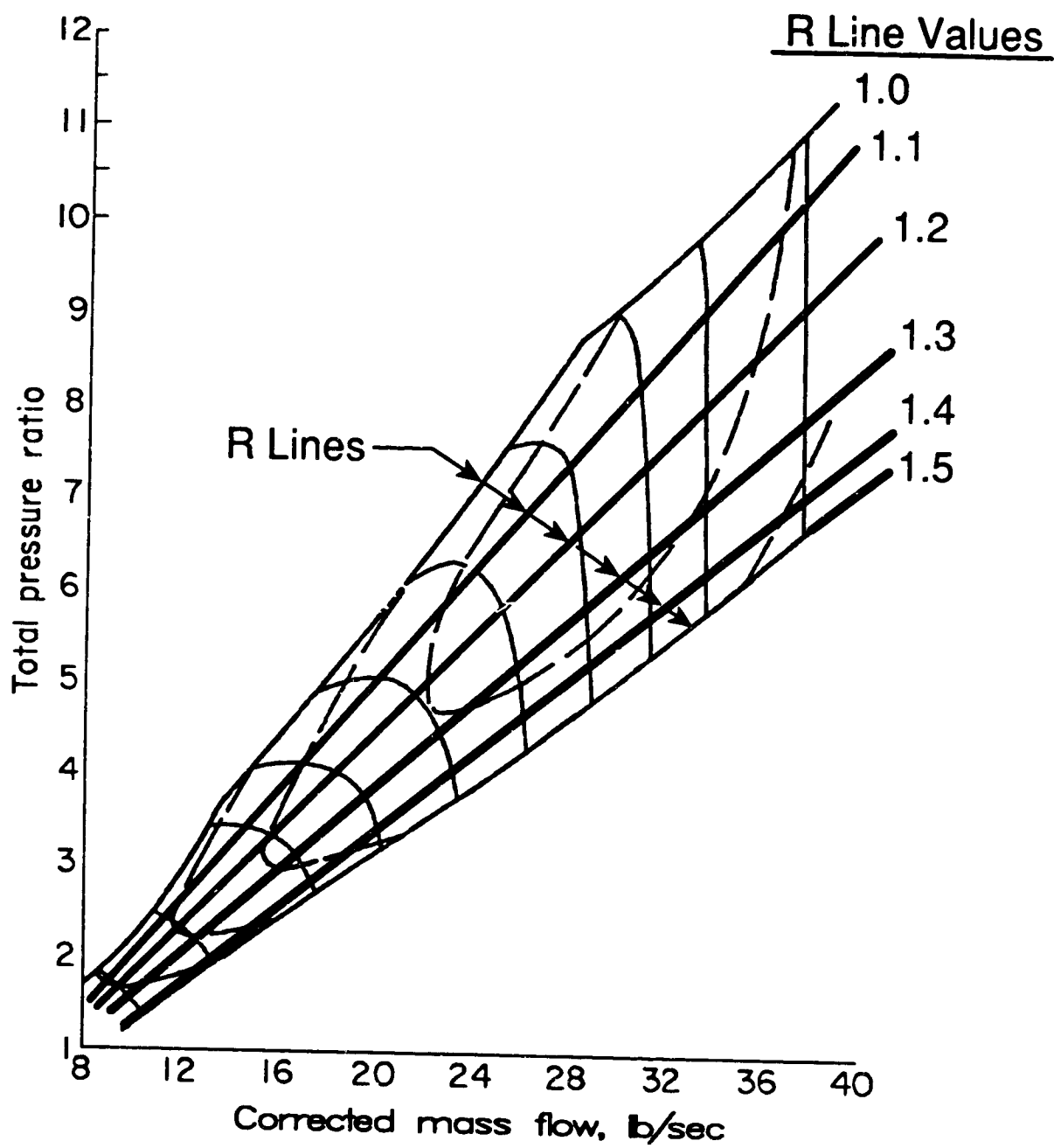


Figure 3. Typical Compressor Performance Map With Arbitrary R Lines

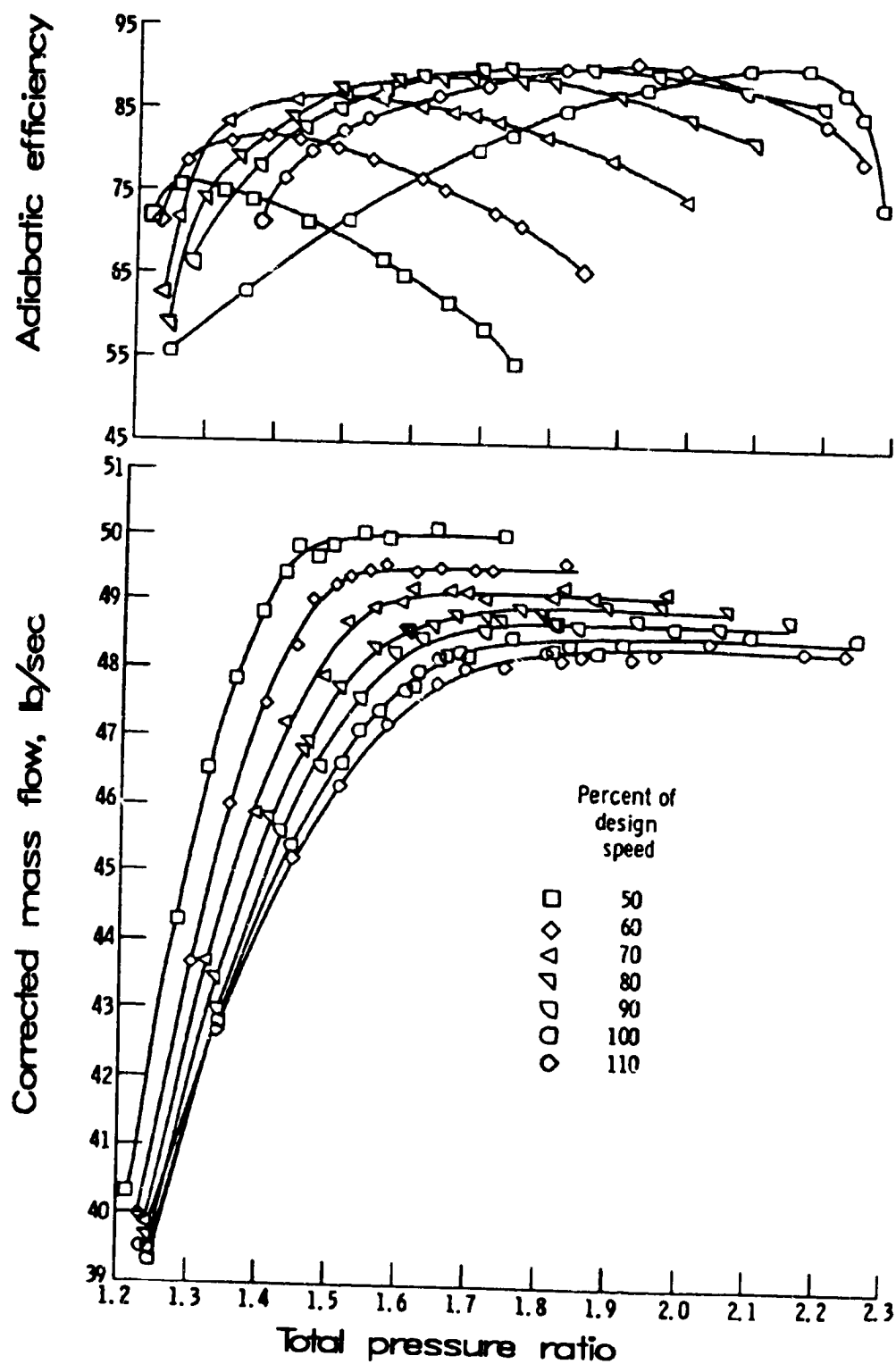


Figure 4. Typical Turbine Performance Map

pressure ratio, the mass flow through this turbine, and user specified design point on the map, the program calculates the turbine map scale factors. The map scale factors for the turbine component have the same definition as the compressor map scale factors. These map scale factors will be used by the program for off-design points to convert the turbine corrected mass flow and efficiency interpolated from the performance maps into the actual performance properties used by the program.

There is a separate program available for generating turbine performance maps (ref. 14). The program is easy to use and can generate turbine performance maps with operating characteristics tailored for the user's application. These generated performance maps are already in the NNEP89 format.

### 3.3 VARIABLE SCHEDULE PERFORMANCE MAPS

Many of the engine component models are already set up to use performance maps for certain input variables. (For example, the inlet recovery data table is set up to use a table of inlet total pressure recovery as a function of inlet referred flow and Mach number.) The format of these data tables is the same as any other NNEP89 data table input, such as the compressor and turbine maps. The user can also input a user-designed data table and use the variable schedule program control component to have the program set any input specification of any engine component as a function of any 3 component inputs, outputs, or engine performance properties. An example how to do this is include in the Section 10.2.16.

#### 4.0 CODE INPUT DESCRIPTIONS

Table 1 shows the layout of an example input dataset for the main inputs to the NNEP89 program. The first record is the title line which helps the user identify the dataset and also appears on the output. The second line is the global &D Namelist input. As with all Namelist data the first character in the field must be blank. This first set of inputs defines how many modes the engine will have, if the user wants to use the chemical equilibrium option, and other similar inputs that define globally what NNEP89 will have to do. A complete list of these inputs are given in Section 10.1. If the chemical equilibrium option has been turned on, the next input is specially for the chemical equilibrium portion of the program. Following the chemical equilibrium inputs (if any), is simply a series of &D Namelist input blocks. These &D input blocks define the engine cycle and how the program should execute the engine cycle. It usually contains only the KONFIG, SPEC and SPCNTL variables. From the global Namelist inputs, the program knows how many modes (NMODES) are to be read. The code reads in the KONFIG, SPEC and SPCNTL variables for all of these modes. Note that most engine configurations only consist of one mode (NMODES=1). After NMODES number of Namelist data have been read, the program will execute the mode which is designated as the design point. Additional design points (with NCODE=3) or a series of off-design points may be executed by inputting additional &D Namelist input blocks.

INPUT FIELD	DESCRIPTION	USAGE
TITLE LINE	1 line input identification	Always required
&D .....&END	Global &D Namelist Inputs	Always required
REACTANTS	CEC switch to read in reactant data	Only input when using the CEC option
(CEC reactants)	Reactant data information	Only input when using the CEC option
(blank line)	Signals end of CEC reactant data	Only input when using the CEC option
THERMO	CEC switch to read in thermo data	Only input when using the CEC option
(CEC THERMO DATA)	Thermo data information	Only input when using the CEC option
END	Signals end of CEC thermo data	Only input when using the CEC option
NAMELIST	CEC switch to read in CEC Namelist data	Only input when using the CEC option
&INPT2 .....&END	CEC Namelist data	Only input when using the CEC option
END (CEC switch to signify end of CEC input data)	Signals end of CEC input data	Only input when using the CEC option
&D MODE=1.....&END	SPEC and KONFIG Namelist inputs for MODE=1	Always required
&D MODE=2.....&END	SPEC and KONFIG Namelist inputs for MODE=2	Only input for engines with 2 or more modes
&D MODE=3.....&END	SPEC and KONFIG Namelist inputs for MODE=3	Only input for engines with 3 or more modes
&D MODE=4.....&END	SPEC and KONFIG Namelist inputs for MODE=4	Only input for engines with 4 or more modes
&D MODE=5.....&END	SPEC and KONFIG Namelist inputs for MODE=5	Only input for engines with 5 or more modes
&D MODE=6.....&END	SPEC and KONFIG Namelist inputs for MODE=6	Only input for engines with 6 modes
&D .....&END	First off-design case or new design point case (these may be repeated as many times as required)	Required for off-design cases or multiple design point cases (set NCODE=3 for multiple design points)

Table 1. Example Input Data Format



## 5.0 NNEP89 ENGINE MODEL DEVELOPMENT

### 5.1 CONFIGURING AN ENGINE

The NNEP89 computer code does not contain any preset engine cycles, such as a single spool turbojet, that the user can simply invoke through a single input. Instead, it requires the user to identify the mechanical and thermodynamic connection between engine components through a set of inputs. In addition, at off-design point operating conditions, the user must identify potential sources of flow mismatch, work imbalance, and rotational speed mismatch within the engine through additional inputs. These mismatches and unbalances are commonly referred to as "errors". Finally, the free variables that will be used to eliminate these "errors" must be specified by the user.

#### 5.1.1 Components and Flow Stations

To use the code, the user must specify all the components which are contained in the desired engine configuration. Unique component numbers are assigned to each of these components. Since the components do not have to be labeled sequentially, flow stations are defined for flows entering and leaving each component. These flow stations tell the code how the components are connected together; the downstream flow station number of one component must correspond to the upstream flow station number of the component that immediately follows it. The restrictions that apply to the assignment of component numbers and flow stations are given in Table 2.

The primary inlet must be always be given a component number of one.

The primary upstream flow station of this inlet must be one.

Component numbers must be in the range of 1 - 199.

Flow station numbers must be in the range of 1 - 99.

Table 2. Input Labeling Restrictions

The thermodynamic properties of the flow are calculated at each of the flow stations through the engine. Table 3 lists these thermodynamic properties and the corresponding Station Property Number which is used to refer to a specific property at any given flow station. Notice that Mach number and static pressure are stored in Station Properties 6 and 7, respectively. However, in general, areas are not calculated by the code. Therefore, Mach number and static pressure are only calculated for a few components. Mach number and static pressure are calculated for the flow stations entering and leaving a mixer. The nozzle component calculates the static conditions for the throat and exit flow stations. The nozzle throat Mach number and static pressure are stored in the flow station location corresponding to the nozzle entrance. The duct (or burner component) has an option to calculate the static conditions for the flow stations entering and leaving that component. The free stream Mach number into the inlet is stored as Station Property 6; however, the free stream static pressure and temperature entering the inlet are stored in Station Properties 2 and 3 (total pressure and temperature). For all other flow stations the static pressure and Mach number are not calculated and will show up as zero on the output.

Station Property Number	Definition
1	Weight flow, lb/s
2	Total pressure, psia
3	Total temperature, °R
4	Fuel-to-air ratio
5	Referred flow, $w\sqrt{T}/P$ Note: corrected flow, $w\sqrt{a}/\delta$ , is printed on the output $w\sqrt{T}/P = 1.5497 w\sqrt{a}/\delta$
6	Mach number
7	Static pressure, psia
8	Interface corrected flow error

Table 3. Station Property Definitions

To aid in assigning the required inputs, a block diagram of the initial engine configuration, as shown in Figure 5, is typically sketched by the user. The component numbers and flow stations are added to this block diagram to aid in visualization of the engine layout and to help in formulation of the input file. The component numbers and flow station numbers in this example do not follow any logical sequence to illustrate that they can be assigned in an arbitrary manner (subject to the restrictions in Table 2). However, it is recommended that when possible the component numbers and flow station numbers be labeled sequentially to aid the user in reading and understanding the results.

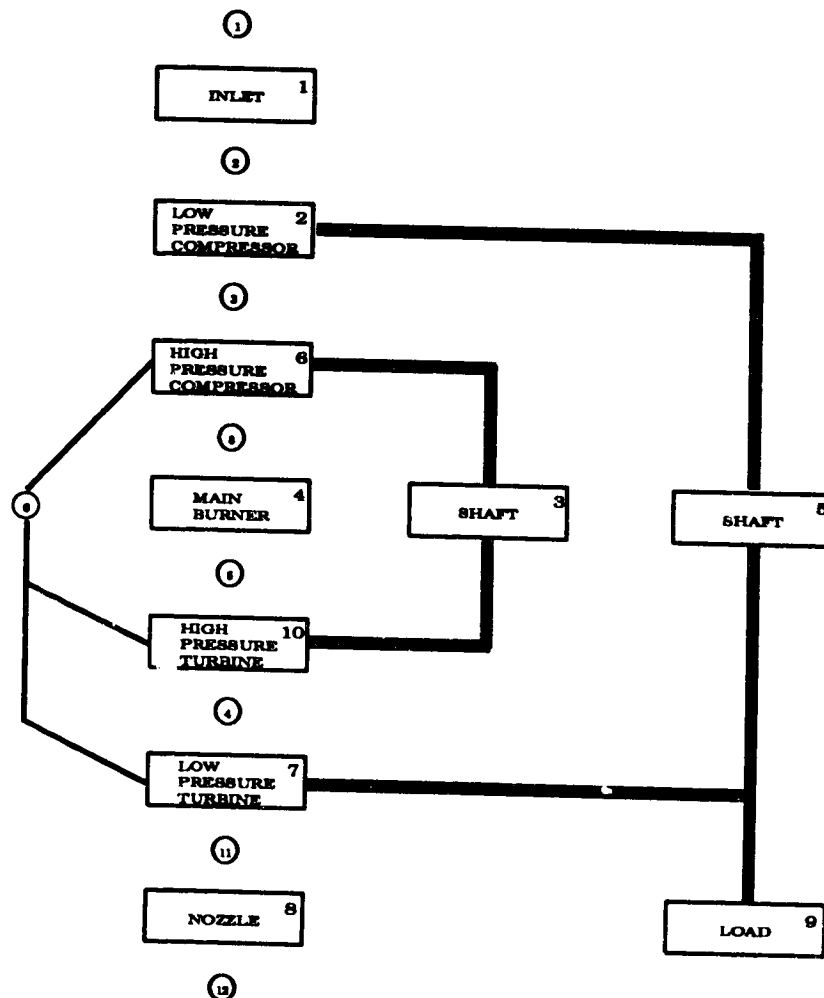


Figure 5. Sample Two-Spool Turbojet

NNEP89 has 16 different components; 11 of them model different engine components, 5 of them tell NNEP89 how to execute the cycle. The engine components are: inlet, duct or burner, water injector or gas generator, compressor, turbine, heat exchanger, splitter, mixer or ejector, nozzle, load or propeller, and shaft. The components that control how NNEP89 executes the cycle are: variable control, variable optimization, variable limit, variable scheduling and conditional control. All components are input using either a unique 4 letter name or a numeric code as shown in Table 4.

The desired engine configuration is entered into NNEP89 through the Namelist variable KONFIG. KONFIG is a two dimensional array of 5 by 200. Each of the components has five values associated with it. The first array element for each component defined is always the quoted string or alternate numeric input listed in Table 4. The quoted string component name input which is a 4 letter name enclosed in single quotes is generally used because of its ease of interpretation. However, some computers do not support the quoted string input, therefore, the alternate numeric input is available. The remaining 4 array locations are used to input the upstream and downstream flow stations or other information pertinent to configuring the engine. The KONFIG variable principally provides the information required to define the

configuration of the desired engine. The KONFIG array location definitions are defined for each component type in Section 10.2. The detailed design information required to define each component is input through the Namelist variable SPEC. SPEC is a two dimensional array of 15 by 200. Each of the components has 15 design input parameters associated with it. These detailed definitions are specified for each component in Section 10.2.

Component Type	Quoted string Input	Alternative Numeric Input
<b>Engine Components</b>		
Inlet	'INLT'	1
Duct or Burner	'DUCT'	2
Water injector	'WINJ'	3
Gas Generator	'GGEN'	3
Compressor or Fan	'COMP'	4
Turbine	'TURB'	5
Heat Exchanger	'HTEX'	6
Flow Splitter	'SPLT'	7
Flow Mixer or Ejector	'MIXR'	8
Nozzle	'NOZZ'	9
Load or Propeller	'LOAD'	10
Shaft	'SHFT'	11
<b>Program Control Components</b>		
Variable Control	'CNTL'	12
Variable Optimization	'OPTV'	13
Variable Limit	'LIMV'	14
Variable Scheduling	'SKED'	15
Conditional Control	'VCNT'	16

Table 4. Input Component Name Definitions

### 5.1.2 Secondary Flow Streams

When configuring an engine that requires a bypass or bleed flow stream, a compressor, duct (not a burner), or flow splitter may be used to create this secondary flow. To create this secondary flow stream the flow station number of the secondary stream is simply input in the fifth array location of the KONFIG variable for that particular component. A secondary flow that is bled off a compressor or a duct must either be bled overboard or it must reenter the main flow stream in a turbine or another duct. A secondary flow that is created by a duct or compressor may not pass through any other components between the point where it is bled and the point where it reenters the engine or is bleed overboard. If the flow is recombined with the main flow stream, the downstream flow station of the compressor or duct where the secondary flow stream is created must be the secondary upstream flow station of the duct or turbine where the flow streams are being recombined. However, all the flow does not have to reenter the engine at one location. It may reenter at multiple ducts and turbines. The momentum of the secondary stream is not accounted for when it is mixed in the duct or turbine with the main flow stream. The momentum of the secondary stream is simply set to the same momentum as the main flow stream at the point they are recombined. However, the enthalpy of the secondary flow stream is accounted for. The temperature of the recombined flow will be determined by the enthalpy of the two streams that are being combined. If overboard bleed is desired, the

secondary flow station is not connected to a duct or a turbine. This flow will not enter into any other calculations and is simply "lost" by the code.

If the user desires to have other components in the secondary flow stream, a flow splitter must be used to create the secondary flow stream. All secondary flow streams that are generated by means of a flow splitter normally will be recombined with the main flow stream in a mixer or an ejector or will exit the engine in a separate nozzle. The secondary flow from a splitter may also reenter the main flow stream through a duct or turbine; however, the momentum of the secondary stream is not accounted for as was discussed above. The secondary flow created by a splitter must either be recombined with the main flow stream or exit through a separate nozzle, it cannot be dumped overboard. If the flow is recombined with the main flow stream, the last downstream flow station of the components in the secondary flow stream must be the secondary upstream flow station of the component where the flow streams are being recombined. Note that splitter/mixer combinations can be nested. In this case the furthest downstream splitter must reenter at the furthest upstream mixer.

## 5.2 MULTIMODE ENGINE CONFIGURATIONS

Modeling of multimode engines is a capability that was developed specifically for NNEP when it was derived from the original Navy NEPCOMP code. This capability allows NNEP89 to model engines that change configuration over various portions of their flight regimes. An example of a multimode engine is a convertible engine for high speed rotorcraft. In the takeoff mode this engine operates as a turboshaft with power being extracted to drive a helicopter type rotor. In the cruise mode the engine operates as a turbofan by disconnecting the helicopter rotor and connecting a fan stage to provide forward thrust. This would be modeled by defining two modes as described below. The first mode would include all the components necessary to represent the turboshaft configuration and a second mode would include all the components necessary to model the turbofan configuration. Most components would be common to both modes.

When setting up a multimode engine, a separate Namelist is used for each mode definition. See Table 1 as well as Sample 5 (Section 12.5) for examples of how to set up the input file. Any component which is common to two modes must have the same component number in both modes. The user is cautioned that if a component appears in more than one mode, the values input in the SPEC variable in the last mode to be input will be the values that are used by the program. Up to six modes may be defined in an engine model, although more than two modes are rarely required. The design point configuration is defined using the MODESN variable. The various modes are executed by setting the variable MODE equal to the number of the mode desired for that particular case.

## 5.3 OFF-DESIGN OPERATION

For design point calculations NNEP89 will automatically ensure continuity of mass, energy and speed between components. For off-design operating points, the continuity of mass and energy are not automatically met. The user must determine which components will have a potential mismatch or "error" in mass flow or energy. A set of free engine variables which is equal to the number of errors in the engine must then be chosen by the user. The variable control component (CNTL) is then used to vary the free variables to reduce the errors within the engine to be smaller than a user specified tolerance. This process will ensure the continuity of mass and energy (within the given tolerance).

As stated above, the first step is to determine where the errors occur within the engine. The continuity of mass will be discussed first. Every component that has its flow dictated by a component map and every component that chokes will only pass a specified amount of corrected

flow,  $w\sqrt{\theta}/\delta$ . For a compressor map the independent variables corrected speed,  $N\sqrt{\theta}$ , and  $R$  uniquely determine the corrected flow through the compressor. However, this does not automatically match the flow being passed to it by the component immediately upstream from it. Similarly, the corrected flow of the turbine will be determined by its pressure ratio and its corrected speed. A nozzle with a fixed throat area will limit the corrected flow to the amount it can pass with Mach one at its throat. Therefore, the entrance to every compressor, turbine and fixed area nozzle will have the possibility of a flow mismatch. The flow coming out of the previous component may not be equal to the amount of flow that the component will pass. These imbalances are commonly called "flow errors".

The second kind of possible mismatch within the engine is a work imbalance. The sum of the energy taken out of the shaft by compressing the air or by driving external loads or propellers may not equal the energy being put into the shaft by the turbines. This type of imbalance is commonly called a "work error". All shafts have potential work errors.

Two components have special operating conditions that introduce unique errors within that component. For the mixer a control component is usually required to force the static pressure of the primary stream to equal the static pressure of the secondary stream. To design a heat exchanger, the input value of temperature rise must be varied until it matches the calculated temperature rise at the design point conditions.

Once all the mismatches or errors are found within the engine, the user must pick an equal number of free variables within the engine. These free variables will be varied by the program in order to obtain error values that are smaller than the tolerance specified by the user. The free variables will depend on the engine configuration, and the particular set of free variables will somewhat affect the operating conditions predicted by the code. A typical list of possible free variables for each component type is given in Table 5.

The variable control component is used to input the engine errors and the free variables into NNEP89. Determining the engine errors, choosing a set of free variables to be used and writing the variable control component inputs is one of the more difficult tasks when using NNEP89, especially for a new or infrequent user. As a result, an automatic control formulation option is available in NNEP89. This option is invoked by setting ACTL to be one or two. The program will then determine the engine errors and will choose a set of free variables to eliminate these errors. All the errors discussed above will be controlled except for the mixer static pressure balance. Because the operation of the mixer is very dependent upon the desired use, the user must input any desired control component for the mixer. Even though the program will assign all the other needed control components, the user should monitor what free variables are being used to ensure the engine is being modeled as the user desires. When executing with this option turned on the code will not try to use free variables that have been previously used in a user written control, nor will it try to eliminate an error that the user has already written a control to eliminate. Thus even if the controls are generally input by the user, the automatic control option may be invoked to ensure that an engine mismatch isn't missed through a user oversight.

The above discussion has shown the necessity of using the variable control component to eliminate mismatches within the engine. However, the use of the control component is much broader than that. It may be used to obtain a specified value for any input or output variable by varying an independent free variable within the engine. One example of this is to force the engine to operate at a specified compressor surge margin. Another example is to obtain a desired thrust level by varying the engine airflow or the burner outlet temperature. These controls may be used at design point, at off-design or both. Variable control components are used extensively in modeling engines with the NNEP89 code.

Component Type	Possible Free Variable
Inlet	Airflow
Compressor or Fan	R Value
	Third Dimensional Value on Stacked Maps
Turbine	Pressure Ratio
	Third Dimensional Value on Stacked Maps
Burner	Burner Outlet Temperature
Mixer or Ejector	Primary or Secondary Flow Area (if variable)
Nozzle	Throat Area
Shaft	Shaft Rotational Speed
Heat Exchanger	Primary Flow Temperature Change
Splitter	Bypass Ratio
Load	Horsepower Extracted

Table 5. Possible Free Variables

Whenever variable control components are used the code must iterate to find a solution that satisfies all the conditions specified by the controls. The number of iterations is normally limited to 50 but may be increased using the MAXNIT input (see section 10.1.2). If any dependent variable is not within its specified tolerance when the maximum number of iterations has been reached, the code will print a warning message on the output and then proceed to the next case.

#### 5.4 REDESIGNING COMPONENTS DURING OFF-DESIGN OPERATION

The IDONE variable can be used to redesign individual components during off-design operation. IDONE is an array that tells the program if a component has been designed or not. If  $IDONE(N)=0$ , component N has not yet been designed. After each point, IDONE is set to one for each component. For off-design points the user can set  $IDONE(N)=0$ , and the program will redesign component N during that execution, based on present conditions. One example for this involves multi-mode engines. After executing the engine in one particular mode, the user switches modes, but this new mode includes another nozzle, component N, that has not yet been designed. The program will set  $IDONE(N)=0$  for this nozzle and will redesign it during that case based on current conditions. However, the user can input the nozzle design parameters directly by setting  $IDONE(N)=1$ .

Setting IDONE=0 for every component will cause the program to redesign all currently-used components. This has the same effect as setting IWAY=1. IWAY is discussed further in Section 10.1.2.

### 5.5 CONFIGURATION LIMITATIONS

Presently, NNEP89 is capable of calculating the steady-state design and off-design thermodynamic performance of engine cycles with as many as 100 flow stations and as many as 200 components. The maximum of 200 components includes engine components and program control components. The user is limited to using a maximum of 50 components of any one type (e.g. maximum of 50 ducts). Note that only 10 variable optimization components may be active at any one time. A maximum of 70 user-supplied component performance maps, with a maximum of 20,000 values may be used.



## 6.0 ENGINE COMPONENTS

As previously stated, NNEP89 has 11 engine components. The engine components are: inlet, duct or burner, water injector or gas generator, compressor, turbine, heat exchanger, splitter, mixer or ejector, nozzle, load or propeller, and shaft. These components will now be discussed in detail. The definition of each input variable for these components is given in Section 10.2. Most of the components contain inputs that are either a fixed variable value or a table reference number. If a table reference number is given, the program will interpolate the user supplied table using the current values of the independent variables of that particular table. A description of these table formats (the dependent and the independent variables) is given in the following discussions. The user must supply the table which relates the program specified dependent and independent variables in a separate map file which gets read in on unit 12.

### 6.1 INLET

The inlet captures the airflow that enters the engine. The inlet conditions can be specified either as temperature and pressure of the airflow or as the geometric or geopotential altitude and Mach number or velocity. If altitude is input, this component will use its internal 1962 ARDC atmospheric model to determine entrance conditions. There is an option to add a delta to the atmospheric model temperature to simulate a non ideal day. The inlet component isentropically compresses the inlet flow to stagnation conditions and then applies a pressure loss using an inlet total pressure recovery factor. The inlet recovery may be input by the user as a constant value or as a table as a function of inlet exit corrected flow and Mach number. If no recovery is input, a default inlet pressure recovery schedule is used as follows:

for Mach  $\leq 1$     Recovery = 1.0

for Mach  $> 1$     Recovery =  $1 - 0.075 (\text{Mach} - 1)^{1.35}$

Inlet airflow (lb/s) can be specified as either corrected airflow at inlet exit, corrected airflow at inlet entrance or actual airflow. The program is preset for two kinds of inlet performance maps. The first is additional inlet drag divided by dynamic pressure as a function of Mach number and inlet exit referred flow. The second is inlet recovery as a function of inlet exit referred flow and Mach number. These maps are used by putting the component map table reference number in the correct SPEC array location.

### 6.2 DUCT or BURNER

A duct can be used as an engine's main combustor, an afterburner, a passage to move flow from one stream and add it to another or a device to model pressure losses. The combustor and the afterburner add fuel to a gas stream to change its temperature. The fuel heating value and combustion efficiency must be input if a burner is desired, unless the user is using the chemical equilibrium thermodynamics model. The desired combustion temperature may be specified and this component will find the fuel to air ratio, or the fuel to air ratio may be specified and the duct component will find the temperature. There are also terms to allow some of the air to be unburned, (e.g. for combustor wall cooling). This unburned air will be added back to the main stream before the burner exit, such that the burner exit temperature will not be equal to the set burner temperature. If the burner cross-sectional area or Mach number is known, the burner will calculate the Rayleigh total pressure loss from the heating of the main stream due to the addition of fuel.

As a passage, a duct can be used as a transition from one component to the next, to bleed off flow for turbine bleed, or bypass flow around other components without using a splitter-mixer combination. When using the duct to bypass flow, the program does not account

for the pressure differences from an entering bypass stream and the main stream. Total enthalpy will be conserved, but the mixed stream will be at the main stream's pressure. The program does not follow the flow path of a duct bypass. Thus, there can not be any other components in-between where the bypass exits the duct and the bypass stream re-enters either a turbine or another duct. If the user wants the program to follow the flow path of a bypass stream (for example to have a bypass go through any one or several component before re-entering in another duct) the splitter components must be used.

The program is preset for three different kinds of user-input duct or burner performance maps; duct or burner pressure drop as a function of the entrance corrected airflow, combustion efficiency as a function of entrance referred airflow and fuel-to-air ratio, and fuel heating value as a function of desired temperature and entrance pressure. These maps are used by inputting the appropriate component map table reference number in the correct SPEC array location.

### 6.3 GAS GENERATOR (Only With Equilibrium Option ON)

The gas generator component is modeled as a rocket sending its exhaust into an engine. It can only be used with the chemical equilibrium option turned on. The gas generator is used to:

- 1) Add a certain amount of a fuel or a fuel and oxidizer stream to an incoming flow, and then determine the total mixture's temperature,
- 2) Add a fuel stream at a certain temperature into an engine, or
- 3) Mix specified fuel and oxidizer streams together for a desired temperature or mixture ratio (like a rocket) into the user's engine. If the user specifies the temperature, the component determines the fuel-to-oxidizer mixture ratio, if mixture ratio is specified, the component will determine the mixture's equilibrium temperature.

The incoming flow stream determines the gas generator's pressure. If the upstream mass flow rate is zero, the user may input the pressure. The user must specify the fuel's composition using the FARRAY array (discussed in Section 10.4). If there is also an oxidizer flow, the user must also specify the oxidizer composition using the OARRAY array (discussed in Section 10.4). Otherwise, the program will assume that there is no oxidizer. Gas generators are commonly added to a cycle by using the splitter component, with the bypass stream as the gas generator input stream. To model a rocket with no upstream flow the user must specify a zero bypass on the splitter. There are no preset performance maps for the gas generator component.

### 6.4 WATER INJECTOR (Only With Equilibrium Option OFF)

The water injector component is used to estimate the effects of water injection into the engine. It can only be used with the chemical equilibrium option OFF. The specific heat, gas constant and the ratio of specific heats for the mixture are accurately accounted for (neglecting chemical dissociation) in the water injector component using a special water-air thermodynamic table. However, downstream the water is accounted as a fuel. The thermodynamic properties are calculated based on the equivalent fuel-to-air ratio. The user can input the amount of water injected (up to 10%, by mass, of the gas flow entering the water injector) and the fraction of water that is vaporized or the user can let the program determine the amount of water necessary for saturation (up to 10%). The program will add the amount of water requested, and calculate the new gas stream temperature. There are no preset performance maps for the water injector component.

## 6.5 COMPRESSOR or FAN

The compressor component can be used for modeling both compressors and fans. At design point, the user specifies the desired pressure ratio, adiabatic efficiency, amount of compressor bleed, and how much compression the compressor bleed obtains. Compressor bleed can be removed at the beginning of the compression process, at the end, or somewhere in-between. The default assumes the bleed is removed at the exit conditions. If the user is using compressor performance maps, the user must tell the program the location of the design point on that map. The map scale factors are set at design point. Compressor performance maps and map scale factors are discussed in more detail in Section 3.1. For off-design points, the compressor component will interpolate from the compressor performance map, using the map scale factors to determine the compressor's operating characteristics.

## 6.6 TURBINE

At design point, the user inputs the desired adiabatic efficiency and, if there are multiple turbines on one shaft, the fraction of total power required by the shaft that will be produced by this turbine. The turbine component will determine the pressure ratio required to produce the necessary amount of power. The user can also set the fraction of turbine cooling used to total bleed flow available. The program can also estimate the amount of compressor bleed air required for turbine cooling at those operating conditions. This option is discussed in reference 9. The turbine cooling flow calculation has been further enhanced by allowing the user to directly input the desired vane and blade temperatures. If the user is using turbine performance maps to predict off-design performance, the user must also specify the turbine design point on that map. At design point, the user inputs will tell the program the desired map design point and the program will determine the map scale factors. Turbine performance maps and map scale factors are discussed in detail in Section 3.2. For off-design points, the turbine component will interpolate from the turbine performance map, using the map scale factors to determine the turbine's operating characteristics.

## 6.7 HEAT EXCHANGER

A heat exchanger transfers thermal energy from one stream to a lower temperature stream. The user inputs the pressure losses for both streams, the heat exchanger effectiveness, and an initial guess the temperature rise of the main stream. The heat exchanger effectiveness is used with the incoming temperatures of the two streams and their relative energies (mass flow\*specific heat) to determine the temperature rise of the main stream. This determines the main stream exit temperature, which determines the amount of thermal energy transferred and the temperature drop of the secondary stream. At design point, NNEP89 must be configured with a Variable Control Component which either varies the heat exchanger effectiveness or varies the guess for the temperature rise of the main stream in order to force the guess value of temperature rise to be equal to the calculated temperature rise. This is used to "design" the heat exchanger. This control is not necessary for off-design points.

The user can also cool the main stream by inputting a negative temperature rise (effectiveness is still a positive number and the secondary stream will be heated). The user can also input the heat exchanger entrance thermodynamic properties for one or both streams when the equilibrium option is used. This could be necessary when the upstream conditions for one or both streams are outside the range able to be calculated by the equilibrium thermodynamic routines.

The program is preset for two different kinds of heat exchanger performance maps. The first is a pressure loss for each incoming stream as a function of its referred mass flow. The second is the heat exchanger effectiveness as a function of actual main gas mass flow and the

ratio of secondary to main mass flows. These maps are used by putting the component map table reference number in the correct SPEC array location.

### **6.8 SPLITTER**

The splitter component is used to create secondary flow paths that will be followed by the program. (See preceding discussions of secondary flows in Section 5.1.2 for the proper use of the splitter component.) For these secondary flows, the program will follow the secondary streams through other components, keeping track of the stream's temperature and pressure. The user inputs to the splitter are the bypass ratio (ratio of secondary exit mass flow to primary exit mass flow) and the total pressure loss of the primary and secondary exit streams. There are no preset performance maps for the splitter component.

### **6.9 MIXER or EJECTOR**

This component combines two streams with possibly different pressures, temperatures, and mass flows. It solves for the exit conditions by simultaneously solving the one-dimensional equations for the conservation of mass, momentum, and energy. The solution of these equations gives two results, a subsonic and a supersonic solution (so named because of the state of the exit stream). The supersonic solution generally results in a higher exit total pressure than the subsonic solution, but it is questionable whether the supersonic solution is physically possible. It is therefore left to the user to choose which solution, subsonic or supersonic, the mixer component will calculate. Mixing losses must be input by the user and are in the form of momentum coefficients for the entering streams or a momentum coefficient for the exit stream. There are options to control mixer inlet and exit areas, mixer design inlet static conditions, and injection angles for the entrance streams. The options are listed, along with some discussion of their use, in Section 10.2.9. There are no preset performance maps for the mixer/ejector component.

### **6.10 NOZZLE**

The nozzle component converts the potential energy of engine exhaust gases into kinetic energy in order to produce useful thrust. The nozzle component can handle both converging and converging-diverging (C-D) nozzles. Traditionally, the converging-diverging nozzle analysis in NNEP and NNEPEQ continuously varied the nozzle exit area such that the flow was fully expanded to ambient static conditions. This method yields the maximum thrust, but it also implies a variable geometry nozzle. The nozzle routine also handles fixed area ratio C-D nozzles. When the exit to throat area ratio is fixed, the nozzle will yield maximum thrust only at its design pressure ratio. If a fixed area ratio nozzle is operating at a pressure ratio less than or greater than its design point pressure ratio, the nozzle is said to be overexpanded or underexpanded, respectively. Due to the unequal exit static and ambient pressures, a pressure thrust or a pressure drag term must be included in the gross thrust calculations. No separation criteria are applied for overexpanded nozzles; hence, the pressure drag term applied may be unusually large. The user also has the option to specify the nozzle exit pressure. The nozzle will expand the nozzle flow to this desired static pressure, regardless of whether the flow is overexpanded or underexpanded. Note that losses due to expansion to other than ambient conditions are categorized as thermodynamic losses. As always, any losses due to viscous effects must be accounted for by velocity coefficients and/or discharge coefficients furnished by the user. These loss coefficients are only applied to the momentum thrust calculations. The nozzle component can also apply divergence (i.e., non-axial) gross thrust losses to converging-diverging nozzles (ref. 7). These losses are accounted for by applying an analytically derived, geometry dependent divergence loss coefficient to the gross momentum thrust.

The fixed area ratio and divergence loss calculations are invoked only when certain inputs are specified. Thus old input files executed with NNEP89 will produce the same results as with previous versions on the code. The nozzle component is preset for two kinds of nozzle performance maps; nozzle flow coefficient as a function of nozzle total to ambient pressure ratio (or if the user inputs a value for nozzle exit static pressure, nozzle flow coefficient as a function of nozzle total pressure to user input exit static pressure ratio) and nozzle velocity coefficient as a function of nozzle total to ambient pressure ratio and the ratio of nozzle exit area to throat area. These maps are used by putting the component map table reference number in the correct SPEC array location.

#### 6.11 LOAD or PROPELLER

This component is used to add or subtract horsepower from an engine cycle. If used as a load, the horsepower subtraction must be input by the user as a negative number or be defined in a component map, with horsepower as a function of actual shaft RPM. These maps are used by inputting the component map table reference number in the correct SPEC array location. If this component is used as a propeller, propeller operating maps can be included. Using engine models with propellers and propeller performance maps is discussed in greater detail in reference 5.

#### 6.12 SHAFTS

The shaft component is for the mechanical connections between compressors, turbines, and loads or propellers. The user must input actual shaft rotational speed, gear ratios between components connected to the shaft and the shaft itself, and the mechanical efficiency between the shaft and each connected component. Four components may be connected to any one shaft. If additional components are required, two or more shafts may be connected together. There are no preset shaft performance maps.

## 7.0 PROGRAM CONTROL COMPONENTS

As previously stated, NNEP89 has 5 program control components. The program control components tell the code how to execute the cycle. These components are variable control, variable optimization, variable limit, variable scheduling, and conditional control. These components will now be discussed in detail. The definition of each input variable for these components is given in Section 10.2.

### 7.1 VARIABLE CONTROL

Since NNEP89 does not have any preset engine configurations, the user must define as input to the program any flow or work errors that the program must drive to zero. The user may also specify that some components operate at a specified condition (such as operating compressors at a constant surge margin). Controls define to the program what independent variables to change to get the engine model to operate in the desired manner. These components are often the most difficult to set up for the user. The philosophy of variable controls was previously discussed in Section 5.3. An option is available to automatically set up the controls at execution time. This option is discussed in detail in reference 8 and an example of its use is given in Section 12.2.

### 7.2 VARIABLE OPTIMIZATION

The optimization component allows the user to maximize or minimize a "cost function" (i.e., the variable being optimized). Through the optimization component, the user tells the program what variable, or variables, to change to maximize or minimize one "cost function". Up to 10 independent variables can be used at one time. The "cost function" can be any performance property or component output in the program (it can not be a station property). Variable optimization is discussed more fully, with an example, in Section 10.2.14.

### 7.3 VARIABLE LIMITER

The variable limiter component tells the program to "watch" certain variables and tell the user if the variables exceed the limits set by the user. If the user is executing with optimization and a limit has been exceeded, the program will penalize the optimization "cost function" (discussed in Section 7.2) to try to get that variable back within desired limits and print out a warning message telling the user that the limit has been exceeded. If optimization is not active and a limit has been exceeded, the program will only print out a warning message that a limit has been exceeded with the rest of the output for that case.

### 7.4 VARIABLE SCHEDULING

Many components are configured to automatically use certain types of performance maps. The user can input these performance maps in a specified tabular format and the program automatically interpolates the tables to obtain the given component performance. A good example of this is the compressor and turbine performance maps. The variable scheduling component is an extension of this performance map capability. Each variable schedule component allows the user to make one input value of any of the 11 engine components (inlets through shafts) to be a function of up to any three other user-specified variables. The value of the specification is determined by interpolating a user supplied performance map table. The table format is the same as for other component map tables.

## 8.0 CHEMICAL EQUILIBRIUM

The default thermodynamic routine used in the program is preset with properties for air and JP4 fuel (carbon-to-hydrogen ratio of 0.5245 and a heating value of 18600 BTU/lb). For many applications, the default thermodynamic routine gives accurate answers. If the user wants to use a fuel other than JP4 fuel, or operate at high-Mach or high-temperature conditions, The default thermodynamic routine may not adequately model these situations. These possibly significant errors in performance estimates and the dissociation effects are discussed in detail in reference 15.

A chemical dissociation option is included with the program, which enables the program to handle just about any combination of chemical species for which thermodynamic data is available. The dissociation option uses a rewritten version of the computer program for calculation of complex Chemical Equilibrium Compositions (CEC) (ref. 16), to calculate thermodynamic properties. The incorporation of the revised CEC program into the NNEP89 program is discussed further in reference 4. For simplicity, user inputs have been kept to a minimum, but still allow a large amount of flexibility for modeling many different types of engine cycles. One limitation to the chemical equilibrium option is that the gas stream will always be at chemical equilibrium. No chemical kinetics effects will be taken into account, such as nozzle gas chemical compositions being frozen at throat conditions. A second limitation to the chemical equilibrium option is that it is only valid in the range of 360 to 9000 °R.

## 9.0 SIMPLIFIED INSTALLATION CALCULATIONS

The NNEP89 code contains some correlations for estimating installation effects. These correlations allow the user to estimate inlet drag or nozzle drag or both. Inlet drags will be estimated if variable SPILL is set equal to TRUE. Nozzle boattail drag will be calculated if variable BOAT is set equal to TRUE. The program will apply the inlet and nozzle drags to calculate the net thrust and total specific fuel consumption (TSFC) with installations drags. The program will also print to the output dataset the calculated inlet and nozzle drags. How to use NNEP89 to estimate each drag will be discussed in the next sections.

### 9.1 INLET DRAGS

The inlet drag, calculated by the code, is the sum of bleed, lip, and spillage drags. Figure 6 shows a side view of a 2-dimension inlet, as an example to help define and explain the inlet installation drag terms.

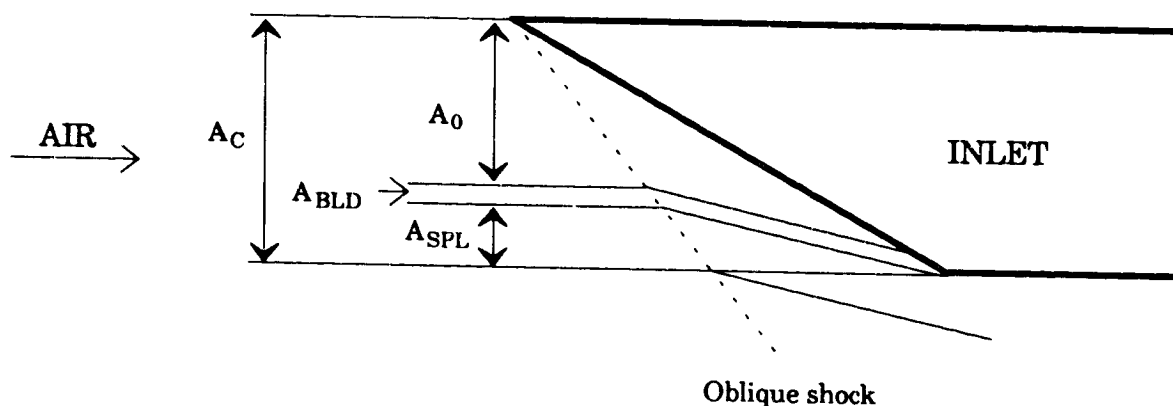


Figure 6. Side View of a 2-D Inlet

$A_C$  denotes the inlet capture area or the total stream tube area for the flow captured or affected by the inlet.  $A_0$  denotes the stream tube area for the airflow entering the engine.  $A_{BLD}$  denotes stream tube area for flow captured by the inlet, but which will be bled or removed from the inlet for stability or performance reasons.  $A_{LIP}$  is not represented in the figure, but it is the stream tube area for the air that impinges on the forward surface of the cowl lip, which will cause a drag.  $A_{SPL}$  denotes the stream tube area for the flow diverted by the inlet. The diversion of this spill flow will cause a drag on the inlet. Each area term can be converted into a mass flow by multiplying by the free stream velocity and density and the momentum for each term can be determined by multiplying its area by the dynamic pressure,  $q$ .

To get each individual inlet drag term, the stream tube area of each term relative to the total stream tube area  $A_C$  is multiplied by that flow's relative change (or loss) of momentum. This is called the drag coefficient,  $C_D$ . The engine airflow, whose stream tube area is denoted by  $A_0$  in Figure 6, has a ram drag which is always calculated by the program as follows:

$$\text{Ram Drag} = \dot{m} * V_0 * 0.031081$$



The drag for each other inlet loss is then determined by an equation similar to the following:

$$DRAG = C_D * A_C * q$$

Bleed drag, whose stream tube area is denoted by  $A_{BLD}$ , is estimated with two terms, the fraction of total capture that will be bled and the change in that flow's momentum. The bleed fraction is estimated by the following equation:

$$f_{BLD} = 0.016 * (\text{Flight Mach number})^{1.5}$$

The change (loss) of momentum is given by the following equation:

$$\Delta M_{BLD} = (2.0 - (\text{Inlet total pressure recovery}) * \sqrt{\text{Flight Mach number}})$$

Both equations together give the following relationships for inlet bleed drag coefficient and inlet bleed drag:

$$C_{DBLD} = f_{BLD} * \Delta M_{BLD}$$

and

$$DRAG_{BLD} = A_C * q * C_{DBLD}$$

Spillage drag, whose stream tube area is denoted by  $A_{SPL}$ , is the total inlet captured airflow minus the engine and inlet bleed airflows. The spillage fraction is given by the following equations:

$$f_{SPL} = \frac{\dot{m}_{SPILL}}{\dot{m}_{TOTAL}} = (1.0 - f_{BLD} - A_0/A_C)$$

with the limitations:

$$\text{if } f_{SPL} > 1.0: f_{SPL} = 1.0$$

$$\text{if } f_{SPL} < 0.0: f_{SPL} = 0.0$$

The change in momentum for the spillage flow is given by one of the following equations:

$$\text{if Mach number} < 1: \Delta M_{SPL} = \frac{(1.33 * (\text{Flight Mach number})^3)}{AMINDS}$$

or

$$\text{if Mach number} > 1: \Delta M_{SPL} = \frac{(1.33 * AMINDS + (\text{Flight Mach number})^2 - 1.0)}{(AMINDS * \text{Flight Mach number})^2}$$

AMINDS is the inlet design Mach number. AMINDS is input by the user, default is 2.7.

The equations for inlet spillage drag coefficient and inlet spillage drag become:

$$C_{D_{SPL}} = f_{SPL} * \Delta M_{SPL}$$

and

$$DRAG_{SPL} = A_C * q * C_{D_{SPL}}$$

The cowl lip drag coefficient is estimated using one of the following equations:

$$\text{if Mach number} < 1: C_{D_{LIP}} = 0.07 - 0.04 * (\text{Flight Mach number})^5$$

or

$$\text{if Mach number} > 1: C_{D_{LIP}} = 0.07 - 0.04 * (\text{Flight Mach number} - 1.0)^{0.25}$$

If spilling, the cowl lip drag is reduced by multiplying  $C_{D_{LIP}}$  by  $(1.0 - f_{SPL})$ . Cowl lip drag is then calculated using the following equation:

$$DRAG_{LIP} = A_C * q * C_{D_{LIP}}$$

Total inlet installation drag is the sum of bleed, spillage, and cowl lip drags. Inlet drags will be calculated if the user sets SPILL equal to TRUE. The user must also input inlet design Mach number (default is Mach 2.7). The user can input the inlet capture area directly through the variable ACAPT or have the program calculate it by setting INLTDS equal to TRUE and inputting a value for the inlet design point spillage fraction in variable SPLDES (default is 0.0). The program will use the present engine parameters, with the inlet design Mach number and inlet design spillage fraction to estimate the capture area. The user may also modify the capture area calculated by the program by inputting a value for variable SIZINL. SIZINL is multiplied by the calculated capture area to get a new, resized capture area. This feature has been found to be useful for quickly determining the effects of changing the inlet design capture area, without changing any other engine or inlet design parameters.

## 9.2 NOZZLE DRAG

Nozzle boattail drag is calculated by the program when variable BOAT is set equal to TRUE. There are no other user inputs for nozzle boattail calculations. Nozzle boattail drag is caused when nozzle exit area is less than the maximum engine nacelle area, as shown in Figure 7.

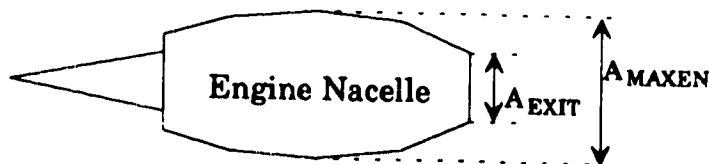


Figure 7. Engine Drawing for Nozzle Drag Calculations

$A_{EXIT}$  is the nozzle exit area and  $A_{MAXEN}$  is the engine nacelle maximum cross-sectional area.  $A_{MAXEN}$  is estimated at design point using the core flow conditions. The program assumes that if there are multiple nozzles,  $A_{EXIT}$  is the sum of all nozzle exit areas and the nozzles are concentric. The boattail drag coefficient,  $C_{D_{BOAT}}$  is calculated using the following equations:

$$A_R = \frac{A_{EXIT}}{A_{MAXEN}}$$

with the limitations:

$$\begin{aligned} \text{if } A_R > 1.0: & \quad A_R = 1.0 \\ \text{if } A_R < 0.05: & \quad A_R = 0.05 \end{aligned}$$

$$\text{if Mach number} < 1: \quad C_{D_{BOAT}} = (1.0 - \sqrt{A_R}) * (0.05 + (\text{Mach number})^{16}) * 0.5$$

$$\text{if Mach number} > 1: \quad C_{D_{BOAT}} = 0.0476 * \frac{(1.0 - A_R)^3}{(\text{Mach number})^{1.5}}$$

The nozzle boattail drag is given by the following equation:

$$DRAG_{BOAT} = A_{MAXEN} * q * C_{D_{BOAT}}$$

NNEP89 assumes that nozzle boattail drag occurs only if the nozzle total exit area is less than the engine nacelle maximum cross-sectional area. If this is not true, if  $A_{EXIT} > A_{MAXEN}$ , nozzle boattail drag is zero.

## 10.0 INPUT/OUTPUT DESCRIPTIONS

The main input data are read from Unit 9. The first input record on Unit 9 is a title line for the job and can be up to 60 characters in length. The remainder of the input is read using FORTRAN Namelist input format with &D as the Namelist block name. Component maps are read from Unit 12. Output is written to Unit 10.

### 10.1 GLOBAL &D NAMELIST INPUTS

The Namelist data follow the title line. The Namelist variables are listed and described below. They are broken out here into five categories only for ease of identification for input by the user.

#### 10.1.1 Input/Output

AMAC	=T Write an output data file in the format required by the AMAC mission program. These data are written to Unit 14 and are appended to the main output file. =F (default) Do not write AMAC data.
DOUTH D	=T Print labels above the DATOUTs on the output. =F (default) Do not print labels.
DRAW	=T Draw a block diagram of engine on output file. =F (default) No diagram will be drawn.
ITERM	=0 (default) No terminal printout. =1 Print at the terminal: altitude, Mach number, inlet recovery, total airflow, gross thrust, fuel flow, net thrust, TSFC and number of iterations as NNEP89 is executing. =2 Same as 1, plus print the number of iterations (NIT) when NIT is a multiple of 5. =-1 Same as +1, except clear terminal screen when number of lines printed out is greater than NUMLIN. =-2 same as +2, except clear terminal screen when number of lines printed out is greater than NUMLIN.
ITPRT	=0 (default) Do not print tables of component maps. =1 Print tables of component maps on output.
LABEL	A control for printing a label at the top of a page to identify the point being executed. Set LABEL=F for the design case. Then, if labels are desired for the off-design cases, set LABEL=T in the off-design case input; then, on a separate line following this Namelist data enter the desired label. (Default value is F.)
LONG	=T (default) Print history of the convergence process. It is advisable to have LONG=T for new problems. =F No printing of convergence history.
MAPLOT	=T Plot scaled component maps, including the operating points. (See reference 6 for more information.) =F (default) No component map plotting.
NUMLIN	Number of lines available on terminal screen. (Used by program to determine when to automatically clear the terminal screen when using ITERM<0. (Default

value is 20.) Using ITERM<0 and NUMLIN is specific to systems that support the FORTRAN call CLRSCR. If your system or terminals do not support this call, remove the CALL CLRSCR line in subroutine CLRVU.

**PINPUT** =T (default) Print Namelist input for a case on the output prior to the results for that case.  
 =F Do not print Namelist input.

**PLOT** =T Enable interactive graphics to draw the engine profile when using the WATE89 program. (See reference 17 for more details.)  
 =F (default) No WATE89 graphics.

**TABLES** =T (default) Component maps are used (Unit 12),  
 =F Component maps are not used

**XNUM** Array of characters that will be used to label the flow station on the output. Each array location can be up to 4 characters input in Hollerith or quoted string format. (Default is the flow station number.)

### 10.1.2 Execution Control

**ACTL** =0 (default) All controls must be defined and activated by the user.  
 =1 Controls to eliminate flow and work errors will be automatically set up and will be activated on the first off-design case. Additionally, required controls when using CALBLD=T or when using a heat exchanger will be automatically activated. Note, because of the many options available, all mixer controls must be input by the user.  
 =2 Same as 1 above, except the user may interactively vary the independent variables chosen to eliminate the flow and work errors.

**ENDIT** =1 Terminate execution after completing the previous case. ENDIT will not appear in the global Namelist inputs, but rather in a later input case where the user wants the program to terminate. (Default is 0.)

**IDONE(i)** Array to tell the program if component i has been designed or not. This is used to redesign a single component during the execution of a set of off design cases. To redesign the entire engine, the user should use IWAY=1 or NCODE=3.  
 = 0 Component i is not designed.  
 = 1 Component i has been designed.

**IWAY** Design point switch. (Default value is 1 for the first point, 0 for following points, and is internally set to -1 for first pass through an engine at design point.) Setting IWAY=1 will execute the next case as a design point, resetting all map scale factors.

**IWT** = 0 (default) Do not execute WATE89.  
 = 1, 2, or 4 Execute WATE89, a weight estimation program. (See reference 10 for more information.)

**MAXNIT** Maximum number of iterations allowed to in order to reach a converged solution. (Default is 50.)

**MODE** Designates the engine mode the code will use for the next case. (Default value is 1.)

**MODESN** Designates which engine mode is the design mode. (Default value is 1.)

**NCASE** When set to 1, this tells the program to reset all arrays and read in a new set of KONFIG and SPEC information. (In essence, clear everything and get ready to start all over again with a new engine configuration and new cycle parameters.)

NCODE       =1 Normal executing.  
               =2 Debug mode (print output after each pass).  
               =-1 or -2 Same as 1 or 2 except full pass through cycle is made on each pass  
               when calculating derivative matrix.  
               =3 Sequence of design points follows (shortens output) and obviates need to  
               supply a &D IWAY=1 &END for each case.  
               =4 Print partial derivative matrix each time it is updated and the values of  
               independent and dependent variables for each control for each iteration.

NMODES       The total number of modes to be configured. (Default value is 1.)

PUNT         =T (default) Use the values of all SPECs from the last converged case as the  
               starting values for the next operating point.  
               =F Use the present values of all SPECs as starting guesses for next point. If  
               the previous case did not converge, some SPECs may have ridiculous values,  
               preventing this new case from converging. Therefore, it is advisable to always  
               have PUNT=T.

#### 10.1.3 Optimization (see Section 10.2.14 for more details)

DEBUG        =0 (default) Do not print the debug information when executing optimization.  
               =1 Print debug information when executing optimization.

NJOPT        Component number which indicates the location of the dependent variable.  
               (Used when optimizing. If 0, the dependent variable is not a DATOUT variable.)

NVOPT        Used when optimizing  
               If NJOPT=0 a value of 1 to 17 indicating which performance property is the  
               dependent variable.  
               If NJOPT≠0 a value of 1 to 9 indicating which DATOUT of component NJOPT  
               is the dependent variable.  
               (NOTE:       NVOPT > 0 minimize dependent variable,  
                           NVOPT < 0 maximize dependent variable.)

TOLOPT       Criteria of convergence of dependent variable. (Used when optimizing, default  
               value is 0.0002.)

#### 10.1.4 Turbine Cooling (see Section 10.2.6 and reference 9 for more details)

CALBLD       =T Perform turbine bleed flow calculations.  
               =F (default) No turbine bleed flow calculations.

ELIFE        Desired engine life. Only used when CALBLD=T for turbine bleed calculations.  
               (Default value is 10000 hours.)

NEWEFF       =T Calculate new turbine efficiency (used for turbine bleed calculations).  
               =F (default) Do not calculate new efficiency.

TMBLAD       Actual desired blade metal temperature, °R. Default is a function of YEARB.

TMVANE       Actual desired vane metal temperature, °R. Default is a function of YEARV.

YEARB        Year of first service of blade. Only used if CALBLD=T for turbine bleed  
               calculations. (Default value is 1990.)

YEARV        Year of first service of vane. Only used if CALBLD=T for turbine bleed  
               calculations. (Default value is 1990.)

#### 10.1.5 Thermodynamic Properties (Including CEC)

C2HRAT	Carbon to hydrogen atomic ratio, used in duct burner calculations to correct the fuel heating value for temperature effects. (Not used if ICEC > 0. Default value is 0.5245035801, the value for JP4.)
FARRAY(i,j)	Array of fuels to be used by the program in DUCTs and GGENs. j is the component number of the DUCT or GGEN. i is an index of reactant number and its relative amount to be used as the fuel, up to 6 components. (FARRAY is only used if ICEC > 0.)
ICEC	= 0 (default) No equilibrium gas properties. = 1 Use chemical equilibrium code (CEC) to calculate thermodynamic properties.
OARRAY	Array (i,j) of oxidizers to be used by the program in GGENs. j is the component number of the GGEN. i is an index of reactant number and its relative amount to be used as the oxidizer, up to 6 components. (OARRAY is only used if ICEC > 0.)
TFUEL	Temperature of fuel added in burner. (Not used if ICEC > 0. Default value is 530°R.)

#### 10.1.6 Installation

ACAPT	Capture area of the inlet, ft <sup>2</sup> . This input is used for the simplified installation calculations. This value will be overridden if INLTDS=T. (Default is 0.0).
AMINDS	Flight Mach number where inlet is designed. Used for simplified installation effects calculations when SPILL=T. (Not to be confused with installation effects calculated by the INSTAL89 program. Default is 2.7.)
BOAT	=T Perform simplified boattail drag calculations. (Not to be confused with boattail drag calculated by the INSTAL89 program.) =F (default) No boattail drag calculations.
INLTDS	=T Specifies this case as the sizing point for inlet area. Used for simplified installation effects. (Not to be confused with installation effects calculated by the INSTAL89 program. Default is FALSE.)
INST	=0 (default) Do not execute INSTAL89. =1 Execute INSTAL89 program.
SIZINL	Scale factor on the inlet capture area. (Default is 1.0).
SPILL	=T Perform simplified installation effects for spillage and lip drag. (Not to be confused with spillage or lip drag calculated by the INSTAL89 program.) =F (default) No spillage or lip drag calculations.
SPLDES	Amount of design spillage when INLTDS=T. (Fraction of inlet airflow that is spilled to total amount of airflow captured. Default is 0.0).

This is all of the data that is read on the global Namelist read. Note that comments may be included anywhere within the Namelist data. These comments must be preceded by a /\* and followed by a \*/. (An example of using this option is given in Section 12.1.) If the ICEC option is used, the chemical equilibrium inputs follow the global Namelist read. Refer to Section 10.4 as

well as reference 4 for complete details on how to set up an input file using the ICEC option. If TABLES=T, the code will go to Unit 12 and read in the maps file. See Section 10.3 for a description of the format for inputting maps.

## 10.2 COMPONENT KONFIG AND SPEC INPUTS AND DATOUT OUTPUTS.

After the code reads the global Namelist inputs, it knows how many modes are to be read. The code reads in the configuration data (KONFIG array) and specifications (SPEC array) for all of these modes. After NMODES number of Namelist data have been read, the program will execute the MODESN as the design point. For all component types except variable control components, the data arrays are read in the following form:

KONFIG(1,N)='NAME',JM1,JM2,JP1,JP2, SPEC(1,N)=V1,V2,V3...V15

where N is the component number and NAME identifies the type of component. The values for NAME, JM1, JM2, JP1, JP2, and the values in the SPEC array are defined in the following sections describing each component. The format is the same as that listed above for variable control components except that the SPCNTL array is used at design point to read in the data.



### 10.2.1 Inlet

#### INPUT

KONFIG(1,N) INLT or 4HINLT or 1  
KONFIG(2,N) Main upstream flow station number (JM1).  
KONFIG(3,N) 0  
KONFIG(4,N) Main downstream flow station number (JP1).  
KONFIG(5,N) 0

SPEC(1,N) Inlet mass flow, lb/sec.  
SPEC(2,N) Free stream temperature, °R. (Only used if SPEC(9,N) or ALTP<0.)  
SPEC(3,N) Free stream static pressure, lb/in<sup>2</sup>. (Only used if SPEC(9,N) or ALTP<0.)  
SPEC(4,N) Inlet additional drag divided by dynamic pressure or table reference number - if blank, additional drag is zero Table = F(x,y) where x = Mach number, y = inlet exit referred flow,  $w\sqrt{T}/P$ .  
SPEC(5,N) Mach number at inlet. If MACH is input it will override SPEC(5,N).  
SPEC(6,N) Inlet recovery, constant or table reference number. If ETAR is input it will override SPEC(6,N).  
=0 Mil Spec is used.  
Mil Spec is: for Mach ≤ 1 Recovery = 1.0  
for Mach > 1 Recovery = 1 - 0.075 (Mach - 1)<sup>1.35</sup>.  
Table = F(x,y) where x = inlet exit corrected weight flow, ( $w\sqrt{\theta}/\delta$ ), y = Mach number.  
SPEC(7,N) Maximum permitted flow in table (if SPEC(6,N)= table reference number).  
SPEC(8,N) Scale factor on corrected weight flow =  $\frac{w\sqrt{\theta}/\delta_{map}}{w\sqrt{\theta}/\delta_{actual}}$  used to read inlet recovery table. (If SPEC(6,N)= table reference number.)  
SPEC(9,N) Geometric altitude, feet. If ALTP is input it will override SPEC(9,N).  
SPEC(10,N) Blank  
SPEC(11,N) If nonzero, SPEC(9,N) is geopotential altitude.  
SPEC(12,N) ΔT to be added to inlet entrance temperature from internal 1962 ARDC atmosphere table. Atmosphere table is used if altitude is input.  
SPEC(13,N) Blank  
SPEC(14,N) Corrected flow at inlet exit. (This input will override SPEC(1,N) and SPEC(15,N).)  
SPEC(15,N) Corrected flow at inlet entrance. (This input will override SPEC(1,N).)

#### OUTPUT

DATOUT(1,N) Inlet ram drag from table or computed, lb.  
DATOUT(2,N) Free stream velocity, ft/sec.  
DATOUT(3,N) Free stream velocity, knots.  
DATOUT(4,N) Ram temperature ratio.  
DATOUT(5,N) Ram pressure ratio.

DATOUT(6,N) Free stream Mach number.  
DATOUT(7,N) Inlet recovery, ratio of exit total pressure to ram pressure.  
DATOUT(8,N) Corrected exit temperature,  $\frac{\text{Inlet exit temperature}}{518.67}$ .  
DATOUT(9,N) Altitude, ft.

Inlet usage notes:

Variables MACH, ALTP, and ETAR, if input, will replace SPECs 5, 9, and 6, respectively, for each inlet in an engine cycle.

If a variable control is used to vary altitude, SPEC(9,N), it is prudent to input a value for SPEC(2,N). When the altitude is greater than 0, the program sets SPEC(3,N)=ambient pressure, and SPEC(2,N) is ignored. If the altitude becomes less than 0, the program uses the value in SPECs 2 and 3 for free stream temperature and pressure. If SPEC(2,N) is left blank, and the altitude becomes less than 0, the program will fail.

## 10.2.2 Duct or Burner

### INPUT

- KONFIG(1,N) 'DUCT' or 4HDUCT or 2
- KONFIG(2,N) Main upstream flow station number (JM1).
- KONFIG(3,N) Secondary upstream flow station (JM2). Used only if duct is used as a passage (no burning).
- KONFIG(4,N) Main downstream flow station number (JP1).
- KONFIG(5,N) Secondary downstream flow station (JP2). Used only if duct is used as a passage (no burning).
- 
- SPEC(1,N)  $\Delta P/P$ , total pressure drop or table reference number. This value is in addition to the optional Rayleigh total pressure drop calculated by the program (see SPEC(2,N)). Table =  $F(x)$  where  $x$  = burner entrance corrected weight flow,  $\frac{w\sqrt{\theta}/\delta_{map}}{w\sqrt{\theta}/\delta_{actual}}$ .
- SPEC(2,N) Design duct entrance Mach number (optional). This is used at the design point to calculate the cross-sectional area of the duct or burner (saved in SPEC(7,N)). This area will be used to calculate the Rayleigh total pressure drop from heating.
- SPEC(3,N) Additional total pressure drop divided by burner inlet referred flow squared.  $\Delta P/P/(W\sqrt{T}/P)^2$ .
- SPEC(4,N)   
 $< 0$  Desired exit fuel-to-air ratio   
 $= 0$  Duct only   
 $> 0$  Desired exit temperature, °R
- SPEC(5,N) Burner efficiency, ratio of actual heat of combustion to ideal heat of combustion, or table reference number (0 if duct). Table =  $F(x,y)$  where  $x$  = burner entrance referred flow,  $w\sqrt{T}/P$ ,  $y$  = fuel-to-air ratio
- SPEC(6,N) Fuel heating value or table reference number, 0 if not burning. (18300 is typical for JP fuel.) - BTU/lb. Table =  $F(x,y)$  where  $x$  = burner exit temperature (SPEC(4,N)),  $y$  = burner entrance pressure.
- SPEC(7,N) Cross-sectional area of duct or burner, in<sup>2</sup>. Calculated at design point if SPEC(2,N) > 0. This area is used to determine duct or burner inlet Mach number. In burners this will also be used to determine Rayleigh pressure drop.
- SPEC(8,N) Ratio of entrance bleed flow to total flow available (duct only).
- SPEC(9,N) Ratio of exit bleed flow to total flow available (duct only),  $\frac{\dot{m}_{JP2}}{(\dot{m}_{JP2} + \dot{m}_{JP1})}$ .
- SPEC(10,N) Fraction of air not heated (bypassed around combustor portion of burner, but will be added back before exiting the duct).
- SPEC(11,N) (Only valid if ICEC=1)   
 $= 0$    
 if SPEC(4,N) > 0 Find fuel-to-air ratio to get desired temperature on fuel-lean side   
 if SPEC(4,N) < 0 Calculate temperature with input fuel-to-air ratio.   
 $= 1$  Find stoichiometric fuel-to-air ratio. If entrance fuel-to-air ratio is less

than this value, add enough fuel to raise the fuel-to-air ratio to the stoichiometric value, otherwise, write error message to output and exit burner.  
 =2 Find fuel-to-air ratio to reach desired temperature, but on the fuel-rich side.

SPEC(12,N) =0 No emission calculations.  
 =1 Calculate emissions index using a simplified empirical correlation based on burner entrance pressure and entrance and exit temperatures. The emissions index is given as grams of oxides of nitrogen produced per kilogram of fuel.  

$$\text{Emissions Index} = 0.0070978 * T_4 * \frac{P_3}{439} * \exp\left(\frac{T_3 - 1471}{345}\right)$$
, where  $P_3$  = burner entrance pressure,  $T_3$  = burner entrance temperature, and  $T_4$  = burner exit temperature.  
 SPEC(13,N) =0 Burner will not reset SPEC(4,N).  
 >0 if SPEC(4,N) <  $T_{JM1}$ , set SPEC(4,N) =  $T_{JM1} + 1$ .  
 SPEC(14,N) Blank  
 SPEC(15,N) Blank

#### OUTPUT

DATOUT(1,N)  $\Delta P/P$ , total pressure drop from Rayleigh calculation (if SPEC(2,N) or SPEC(7,N) was specified at design point).  
 DATOUT(2,N)  $\Delta P/P$ , total pressure drop specified by SPEC(1,N) + SPEC(3,N).  
 DATOUT(3,N) Design duct entrance Mach number, equal to SPEC(2,N).  
 DATOUT(4,N) Ratio of fuel mass flow to duct inlet mass flow.  
 DATOUT(5,N) Duct cross-sectional area, in<sup>2</sup>.  
 DATOUT(6,N) Fuel mass flow, lb/hour.  
 DATOUT(7,N) Duct entrance Mach number (if SPEC(2,N) or SPEC(7,N) is specified at design point).  
 DATOUT(8,N) Burner efficiency (equal to SPEC(5,N) if ICEC=0, 0 if no burning, 1 if burning and ICEC=1).  
 DATOUT(9,N) Burner outlet temperature before fraction of air not heated (SPEC(10,N) is added back to the burner flow, °R.

#### Duct or burner usage notes:

When using the duct to bypass flow, the program does not account for the pressure differences from an entering bypass stream and the main stream. This means that the bypass stream's pressure is only used to calculate its enthalpy, total enthalpy will be conserved, but the mixed stream will be at the main stream's pressure.

The duct or burner component can calculate the Mach number and static conditions for the incoming flow if a Mach number (at design-point only) or a cross-sectional area is input. Previously, this component would only calculate static conditions if combustion occurred. The static pressure and Mach number for both the main upstream and downstream flows are included in the station property output.

SPEC(13,N) should be used when a variable control varies the burner temperature. SPEC(13,N) will keep the burner temperature from dropping below the burner entrance

temperature. This situation could cause convergence problems with the program.

### 10.2.3 Gas Generator (only used when ICEC=1)

#### INPUT

KONFIG(1,N) 'GGEN' or 4HGGEN or 3  
KONFIG(2,N) Main upstream flow station number (JM1).  
KONFIG(3,N) 0  
KONFIG(4,N) Main downstream flow station number (JP1).  
KONFIG(5,N) 0

SPEC(1,N) Exit temperature of fuel or fuel and oxidizer stream, °R.  
SPEC(2,N) Fuel to oxidizer mass ratio (default = 0.3). Since the fuel-to-oxidizer mass ratio is double valued with respect to combustion temperature, an initial guess is necessary.  
SPEC(3,N) Gas generator pressure, psia (default is pressure at JM1).  
SPEC(4,N) Fuel mass flow rate, lb/s.  
SPEC(5,N) Oxidant mass flow rate, lb/s.  
SPEC(6,N) Fraction of fuel not included in engine fuel consumption calculations.  
SPEC(7,N) Fraction of oxidizer not included in engine fuel consumption calculations.  
SPEC(8,N)-  
through-  
SPEC(15,N) Blank

#### OUTPUT

DATOUT(1,N) Gas generator exit temperature, °R.  
DATOUT(2,N) Assigned gas generator fuel-to-oxidant mass ratio.  
DATOUT(3,N) Gas generator pressure, psia.  
DATOUT(4,N) Gas generator fuel mass flow, lb/s.  
DATOUT(5,N) Gas generator oxidant mass flow, lb/s.  
DATOUT(6,N) Total fuel and oxidizer mass flow included in engine fuel consumption calculations, lb/hr.  
DATOUT(7,N) Calculated fuel-to-oxidant mass ratio.  
DATOUT(8,N) Total propellant mass flow, lb/hr.  
DATOUT(9,N) Total fuel and oxidizer mass flow not included in engine fuel consumption calculations, lb/hr.

#### Gas generator usage notes:

The gas generator component is modeled as a burner for which the user can specify the actual amount of fuel (and oxidizer) added into a gas stream or as a rocket injector and combustion chamber sending its exhaust into an engine. The user can use it to:

- 1) Add a certain amount of a fuel or a fuel and oxidizer stream to an incoming flow,
- 2) Add a fuel stream at a certain temperature into an engine, or

3) Mix specified fuel and oxidizer streams together for a desired temperature or mixture ratio (like a rocket injector and combustion chamber) into the user's engine.

The incoming flow stream determines the gas generator's pressure. If an incoming stream does not exist the user must input the pressure. The user specifies in the FARRAY array which reactants and their relative amounts the program will use as the fuel. If there is also an oxidizer, the user must specify the reactants and their relative amounts in the OARRAY array. A discussion and example using a gas generator is included in Section 10.4.2. The gas generator must have at least a FARRAY array to operate; an OARRAY is optional depending on the option specified by the user.

Gas generators are commonly added to a cycle by using a splitter, with the bypass stream as the gas generator input stream. To model a rocket, flow the user must specify a zero bypass on the splitter and exhaust the gas generator flow into a nozzle.

There are five input combinations depending on the input information the user supplies:

Input Combination 1:

User supplies FARRAY, OARRAY, temperature, a starting guess for the fuel-to-oxidant ratio (see SPEC(2,N)), and a flow rate for either fuel or oxidizer. The routine will calculate the fuel to oxidant mass ratio required to get the desired temperature.

Input Combination 2:

User supplies FARRAY, OARRAY, no temperature (or 0), and either a fuel to oxidant mass ratio and a flow rate for the fuel or oxidizer; or no fuel to oxidant mass ratio and both the flow rates for the fuel and oxidizer. The routine combines the fuel and oxidizer in the specified amounts and calculates the exit temperature.

Input Combination 3:

User supplies FARRAY, no OARRAY, temperature, and a flow rate for the fuel (with no OARRAY given, the routine ignores any fuel to air ratio or oxidizer flow rate input). The routine then calculates the fuel's thermodynamic properties at the specified temperature and pressure.

Input Combination 4:

User supplies same information as Input Combination 2, except there is incoming flow. The user must input an oxidizer flow rate if one is desired (fuel to oxidizer mass ratio is ignored). The routine then adds the specified amounts of fuel and oxidizer to the incoming flow and calculates the mixture's thermodynamic properties.

Input Combination 5:

User supplies same information as Input Combination 3, except there is incoming flow. The routine then adds the specified amount of fuel, at the specified temperature, to the incoming flow and calculates the mixture's thermodynamic properties.

#### 10.2.4 Water Injector (only used when ICEC=0)

##### INPUT

KONFIG(1,N) 'WINJ' or 4HWINJ or 3  
KONFIG(2,N) Main upstream flow station number (JM1).  
KONFIG(3,N) 0  
KONFIG(4,N) Main downstream flow station number (JP1).  
KONFIG(5,N) 0

SPEC(1,N) Water-to-air mass flow ratio (maximum value is 0.10).  
SPEC(2,N) Fraction of water vaporized.  
SPEC(3,N)  $\Delta P/P$ , total pressure drop.  
SPEC(4,N) Saturation switch: If 0, use SPEC(1,N); if 1, saturate.  
SPEC(5,N)-  
through-  
SPEC(15,N) Blank

##### OUTPUT

DATOUT(1,N) Actual water-to-air mass flow ratio.  
DATOUT(2,N) Input value of fraction of water vaporized.  
DATOUT(3,N) Saturation value of water-air mixture.  
DATOUT(4,N) Actual fraction of water vaporized.  
DATOUT(5,N)  $\Delta T$ , change in temperature ( $^{\circ}R$ ).  
DATOUT(6,N) Water mass flow rate ( $lb_m/hr$ ).  
DATOUT(7,N)  $\Delta P/P$ , total pressure drop.  
DATOUT(8,N) Blank  
DATOUT(9,N) Blank

##### Water injector usage notes:

To turn on the water injector, SPEC(1,N) must be nonzero. The input value of SPEC(1,N) will be used unless SPEC(4,N) is equal to unity in which case SPEC(1,N) will be over-ridden by the saturation value. The saturation value is computed internally and is limited to a 0.10 water-to-air mass ratio.

To turn off the water injector, set SPEC(1,N) to zero. Even though SPEC(4,N) may be equal to unity (i.e., saturation), no water will be injected.



### 10.2.5 Fan or Compressor

#### INPUT

- KONFIG(1,N) 'COMP' or 4HCOMP or 4
- KONFIG(2,N) Main upstream flow station number (JM1).
- KONFIG(3,N) 0
- KONFIG(4,N) Main downstream flow station number (JP1).
- KONFIG(5,N) Bleed flow station for the compressor (JP2).
- 
- SPEC(1,N) R value used to read compressor map tables.
- SPEC(2,N) Ratio of compressor bleed flow to total flow into compressor.
- SPEC(3,N) Scale factor on corrected speed =  $\frac{N/\sqrt{\theta}_{actual}}{N/\sqrt{\theta}_{map}}$  (usually input as 1).
- SPEC(4,N) Corrected weight flow ( $w\sqrt{\theta}/s$ ) or table reference number. Table =  $F(x,y,z)$  where  $x = R$  value,  $y =$  corrected speed ( $N/\sqrt{\theta}$ ),  $z =$  3rd dimensional argument value.
- SPEC(5,N) Scale factor on corrected weight flow =  $\frac{w\sqrt{\theta}/s_{actual}}{w\sqrt{\theta}/s_{map}}$  (usually input as 1).
- SPEC(6,N) Compressor adiabatic efficiency or table reference number. Table =  $F(x,y,z)$  where  $x = R$  value,  $y =$  corrected speed ( $N/\sqrt{\theta}$ ),  $z =$  3rd dimensional argument value.
- SPEC(7,N) Scale factor on compressor efficiency =  $\eta_{actual}/\eta_{map}$  (usually input as 1).
- SPEC(8,N) Compressor pressure ratio or table reference number. Table =  $F(x,y,z)$  where  $x = R$  value,  $y =$  corrected speed ( $N/\sqrt{\theta}$ ),  $z =$  3rd dimensional argument value.
- SPEC(9,N) Scale factor on pressure ratio =  $\frac{PR_{actual} - 1}{PR_{map} - 1}$  (usually input as 1).
- SPEC(10,N) 3rd dimensional argument value used to read compressor map tables.
- SPEC(11,N) Fractional bleed horsepower loss due to interstage bleed.
- SPEC(12,N) Desired adiabatic efficiency at specified R and  $N/\sqrt{\theta}$  at design point. (Used to determine map scale factor for compressor efficiency.)
- SPEC(13,N) Desired pressure ratio at specified R and  $N/\sqrt{\theta}$  at design point. (Used to determine map scale factor for compressor pressure ratio.)
- SPEC(14,N) Corrected speed ( $N/\sqrt{\theta}$ ) at design point on maps.
- SPEC(15,N) Blank

#### OUTPUT

- DATOUT(1,N) Power required by compressor (negative), hp.
- DATOUT(2,N) Angular velocity of the compressor, rpm.
- DATOUT(3,N) 3rd dimensional argument value used for compressor map tables.
- DATOUT(4,N) R value used for compressor map tables.
- DATOUT(5,N) Surge margin (percent) =  $\left[ \frac{W_{corr}/W_{corr\_surge}}{PR/PR_{surge}} - 1 \right] \times 100.$

DATOUT(6,N) Corrected speed ( $N\sqrt{\Theta}$ ) used for compressor map tables.  
DATOUT(7,N) Scale factor on corrected weight flow, ( $w\sqrt{\Theta}/s$ ).  
DATOUT(8,N) Compressor efficiency.  
DATOUT(9,N) Compressor pressure ratio.

Fan or compressor usage notes:

The format for the compressor map tables are included in Section 10.3. The program has the capability to plot the scaled component maps and the actual operating points on these maps. This capability is described in reference 6.

If performance maps are used, a variable control is required for off design operation of the program to drive the flow error on the incoming gas stream to zero.

### 10.2.6 Turbine

#### INPUT

- KONFIG(1,N) 'TURB' or 4HTURB or 5
- KONFIG(2,N) Main upstream flow station number (JM1).
- KONFIG(3,N) Bleed flow station (JM2) (Matches bleed flow station for the corresponding compressor).
- KONFIG(4,N) Main downstream flow station number (JP1).
- KONFIG(5,N) 0
- 
- SPEC(1,N) Pressure ratio used to read turbine map tables.
- SPEC(2,N) Ratio of total bleed into turbine to total bleed available.
- SPEC(3,N) Scale factor on corrected speed =  $\frac{N/\sqrt{\theta_{actual}}}{N/\sqrt{\theta_{map}}}$  (usually input as 1).
- SPEC(4,N) Corrected weight flow ( $W\sqrt{\theta}/\delta$ ) or table reference number.  
Table =  $F(x,y,z)$  where  $x$  = pressure ratio,  $y$  = corrected speed ( $N/\sqrt{\theta}$ ),  
 $z$  = 3rd dimensional argument value.
- SPEC(5,N) Scale factor on corrected weight flow =  $\frac{W\sqrt{\theta}/\delta_{actual}}{W\sqrt{\theta}/\delta_{map}}$  (usually input as 1).
- SPEC(6,N) Turbine adiabatic efficiency or table reference number. Table =  $F(x,y,z)$  where  
 $x$  = pressure ratio,  $y$  = corrected speed ( $N/\sqrt{\theta}$ ),  $z$  = 3rd dimensional argument  
value.
- SPEC(7,N) Scale factor on turbine efficiency =  $\eta_{actual}/\eta_{map}$  (usually input as 1).
- SPEC(8,N) Scale factor on turbine pressure ratio =  $\frac{PR_{actual}^{-1}}{PR_{map}^{-1}}$  (usually input as 1).
- SPEC(9,N) Ratio of turbine bleed flow injected at the turbine entrance to the total bleed  
flow into this turbine. The remainder of the bleed flow into the turbine will be  
added at the turbine exit.
- SPEC(10,N) 3rd dimensional argument value used to read turbine map tables.
- SPEC(11,N) Desired adiabatic efficiency at design point.
- SPEC(12,N)  $N/\sqrt{\theta}$  at design point on maps.
- SPEC(13,N) Design point turbine horsepower split (usually set equal to 1). For multiple  
turbines on one shaft, what fraction of the total shaft work required this  
turbine must produce.
- SPEC(14,N) Cooling type code (only used if CALBLD=T, see next page for code definitions).
- SPEC(15,N) Number of turbine stages (only used if CALBLD=T, see next page).

#### OUTPUT

- DATOUT(1,N) Power produced by the turbine (positive), hp.
- DATOUT(2,N) Angular velocity of the turbine, rpm.
- DATOUT(3,N) 3rd dimensional argument value used for turbine map tables.
- DATOUT(4,N) Turbine pressure ratio used for turbine map tables.
- DATOUT(5,N) Scale factor on corrected speed,  $N/\sqrt{\theta}$ .

DATOUT(6,N) Corrected speed ( $N/\sqrt{\theta}$ ) used for turbine map tables.  
DATOUT(7,N) Scale factor on corrected weight flow  $w\sqrt{\theta}/s$ .  
DATOUT(8,N) Turbine efficiency.  
DATOUT(9,N) Turbine overall pressure ratio.

Turbine usage notes:

The format for the turbine map tables are included in Section 10.3. The program has the capability to plot the scaled component maps and the actual operating points on these maps. This capability is described in reference 6.

If performance maps are used, a variable control is required for off design operation of the program to drive the flow error on the incoming gas stream to zero.

# Turbine cooling bleed usage notes:

CALBLD is set to T when bleed requirements are to be determined. See reference 9 for a complete discussion of the turbine cooling analysis. A control must be set to vary SPEC(2,N) of the compressor in which the bleed is being removed to drive PERF(15) to zero. Other controls may be turned on as required.

SPEC(14,N) of the turbine is set to indicate the type of cooling row by row. The default value is 88 (one stator row and one rotor row with full film cooling). See Table 6 for the definitions of cooling types.

SPEC(15,N) of the turbine is the number of turbine stages and is used only in sizing bleed requirements. Default is 1 stage.

ELIFE = Desired engine life. Default is 10000 hrs.

NEWEFF = Calculate new turbine efficiency due to cooling. Default is F.

TMBLAD = Actual desired blade metal temperature, °R. Default is a function of YEARB.

TMVANE = Actual desired vane metal temperature, °R. Default is a function of YEARV.

YEARB = Rotor blade first year of service. Used to determine blade temperature. Default is 1990.

YEARV = Stator vane first year of service. Used to determine vane temperature. Default is 1990.

For all other cases after sizing the bleed, the user normally sets SPEC(9,N) of the bleed control to zero and CALBLD=F.










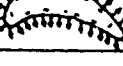
Cooling Code		Definition
0		Uncooled
1		Convection
2		Convection with coating
3		Advanced Convection
4		Film with Convection
5		Film with Convection
6		Film with Convection
7		Transpiration with Convection
8		Full cover film
9		Transpiration

Table 6. Cooling Type Definitions.

### 10.2.7 Heat Exchanger

#### INPUT

- KONFIG(1,N) 'HTEX' or 4HHTEX or 6  
KONFIG(2,N) Upstream flow station number for the flow stream being heated (JM1).  
KONFIG(3,N) Upstream flow station number for the flow stream being cooled (JM2).  
KONFIG(4,N) Downstream flow station number for the flow stream being heated (JP1).  
KONFIG(5,N) Downstream flow station number for the flow stream being cooled (JP2).
- SPEC(1,N)  $\Delta P/P$  or table reference number for heated flow stream (JM1).  
Table =  $F(x)$  where  $x$  = Referred flow of JM1 stream,  $w\sqrt{T}/P$ .  
SPEC(2,N)  $\Delta P/P$  or table reference number for cooled flow stream (JM2).  
Table =  $F(x)$  where  $x$  = Referred flow of JM2 stream,  $w\sqrt{T}/P$ .  
SPEC(3,N) Guess value for temperature rise of main stream (JM1), °R.  
SPEC(4,N) Heat exchanger effectiveness (ratio of actual thermal energy transfer to ideal thermal energy transfer) or table reference number.  
Table =  $F(x,y)$  where  $x$  = weight flow of the stream being heated,  $W_{JM1}$ ,  $y$  = weight flow ratio of the stream being heated to the stream being cooled,  $\frac{W_{JM2}}{W_{JM1}}$ .  
SPEC(5,N) Scale factor on heat exchanger effectiveness.

SPECs 6 through 13 are only active if ICEC=1 (see Heat exchanger usage notes).

- SPEC(6,N) Specific enthalpy for heated flow stream (JM1), BTU/lb.  
SPEC(7,N) Ratio of specific heats (gamma) for heated flow stream (JM1).  
SPEC(8,N) Gas constant (R) for heated flow stream (JM1), BTU/lb-°R.  
SPEC(9,N) Specific enthalpy for cooled flow stream (JM2), BTU/lb.  
SPEC(10,N) Ratio of specific heats (gamma) for cooled flow stream (JM2).  
SPEC(11,N) Gas constant (R) for cooled flow stream (JM2), BTU/lb-°R.  
SPEC(12,N) Temperature for heated flow stream (JM1), °R.  
SPEC(13,N) Temperature for cooled flow stream (JM2), °R.  
SPEC(14,N) Blank  
SPEC(15,N) Blank

#### OUTPUT

- DATOUT(1,N)  $\Delta P/P$ , total pressure loss for heated flow stream (JM1), equal to SPEC(1,N).  
DATOUT(2,N)  $\Delta P/P$ , total pressure loss for cooled flow stream (JM2), equal to SPEC(2,N).  
DATOUT(3,N) Blank  
DATOUT(4,N) Heat exchanger effectiveness, equal to SPEC(4,N).  
DATOUT(5,N) Scale factor on effectiveness, equal to SPEC(5,N).  
DATOUT(6,N) Calculated temperature rise of heated flow stream (JM1), °R.

- DATOUT(7,N) Temperature rise of heated stream divided by the difference between the entrance temperatures of the heated and cooled streams.  $\frac{(T_{JP1}-T_{JM1})}{(T_{JM1}-T_{JM2})}$ .
- DATOUT(8,N) Temperature rise difference, ((guess value/calculated value)-1).
- DATOUT(9,N) Calculated temperature rise of cooled stream (JM2), °R.

#### Heat exchanger usage notes:

At design point, there must be a control to vary either the estimate for the heat exchanger effectiveness or the temperature rise of the main stream such that these two SPECS are consistent, (DATOUT(8,N)=0). This is used to "design" the heat exchanger. Usually the other controls are turned off, so as not to change other cycle parameters while designing the heat exchanger. For off-design points, this control is turned off.

The heated and cooled streams can be reversed without reconfiguring the engine by inputting a negative temperature rise (SPEC(3,N) < 0).

SPECS 6 through 13 have been added to override the calling of the CEC routines to get thermodynamic properties for one or both of the upstream flows (either JM1 or JM2 stream). This feature is useful if the upstream conditions for one or both streams are outside the range of the CEC routines. This allows the user to input the thermodynamic properties for that stream. The CEC calls are overridden for the JM1 (main stream) if SPEC(7,N) > 0. The CEC calls are overridden for the JM2 (secondary stream) if SPEC(10,N) > 0.

### 10.2.8 Flow Splitter

#### INPUT

KONFIG(1,N) 'SPLT' or 4HSPLT or 7  
KONFIG(2,N) Main upstream flow station number (JM1).  
KONFIG(3,N) 0  
KONFIG(4,N) Main downstream flow station number of the split flow (JP1).  
KONFIG(5,N) Secondary downstream flow station number of the split flow (JP2).  
  
SPEC(1,N) Bypass ratio, ratio of secondary downstream mass flow to main downstream  
mass flow,  $\frac{\dot{m}_{JP2}}{\dot{m}_{JP1}}$ .  
SPEC(2,N)  $\Delta P/P$  of the main flow stream (JP1).  
SPEC(3,N)  $\Delta P/P$  of the secondary (bypass) flow stream (JP2).  
SPEC(4,N)-  
through-  
SPEC(15,N) Blank

#### OUTPUT

DATOUT(1,N) Bypass ratio, ratio of secondary downstream mass flow to main downstream  
mass flow,  $\frac{\dot{m}_{JP2}}{\dot{m}_{JP1}}$ .  
DATOUT(2,N)  $\Delta P/P$  in the main flow stream (JP1).  
DATOUT(3,N)  $\Delta P/P$  in the secondary (bypass) flow stream (JP2).  
DATOUT(4,N) Total weight flow of primary stream (JM1), lb/s.  
DATOUT(5,N) Sum of weight flows of reactants of primary stream (JM1), lb/s.  
DATOUT(6,N)-  
through-  
DATOUT(9,N) Blank

#### Flow splitter usage notes:

Each splitter bypass stream must eventually end in a nozzle or mixer. If such is not the case, use a DUCT instead of a splitter, setting SPEC(9,N) of the DUCT to the desired bypass ratio. The program also follows the secondary (bypass) stream through other components (if any) and keeps track of the bypass stream's pressure (as opposed to how the program handles bypass streams from DUCTs). This is discussed in Section 5.1.2.



### 10.2.9 Mixer or Ejector

#### INPUT

KONFIG(1,N)	'MIXR' or 4HMIXR or 8
KONFIG(2,N)	Main upstream flow station number (JM1) if SPEC(7,N) = 0. Ejector secondary flow stream if SPEC(7,N) > 0.
KONFIG(3,N)	Secondary upstream flow station number (JM2) if SPEC(7,N) = 0. Ejector primary flow stream if SPEC(7,N) > 0.
KONFIG(4,N)	Main downstream flow station number (JP1).
KONFIG(5,N)	0
SPEC(1,N)	Mixer inlet area of JM1 stream, in <sup>2</sup> . Calculated at design point using SPEC(3,N) and design point operating conditions.
SPEC(2,N)	Mixer inlet area of JM2 stream, in <sup>2</sup> . Calculated at design point using SPEC(3,N) and design point operating conditions.
SPEC(3,N)	(Required at design point to determine mixer inlet areas.) < 1 Mach number of JM1 stream at design point. > 1 Total to static pressure ratio of JM1 stream at design point.
SPEC(4,N)	If SPEC(4,N) > 0 momentum coefficient for mixer exit stream (JP1); ignore SPEC(13,N) and SPEC(14,N).
SPEC(5,N)	=0 Independent mixer inlet areas. =1 Total mixer inlet area is held constant, any change in the secondary (JM2) area has a corresponding change in the primary (JM1) area.
SPEC(6,N)	=0 Inactive. =1 (Active at design point only). SPEC(3,N) will be the minimum total to static pressure ratio or minimum Mach number of the stream that has the lowest total pressure (either JM1 or JM2).
SPEC(7,N)	=0 Mixer will mix two subsonic streams. =1 Supersonic mixer. At design point, expand the JM2 stream to the same static pressure as the JM1 stream. Off-design, do the same calculation as SPEC(7,N) = 2. =2 Do ejector calculation (JM2 stream is supersonic). (See mixer usage notes.)
SPEC(8,N)	= 0 Subsonic solution to mixing equations. = 1 Supersonic solution to mixing equations.
SPEC(9,N)	If SPEC(7,N) > 0, design point ejector primary (JM2 stream) Mach number (typically around 1.5 to 3), otherwise blank.
SPEC(10,N)	Mixer area ratio = $\frac{\text{Mixer exit area}}{\text{Total mixer inlet area}}$
SPEC(11,N)	Injection angle (degrees) for JM1 if SPEC(7,N) = 0, JM2 if SPEC(7,N) > 0. (Default is 0.0)
SPEC(12,N)	Factor used to check for over-expansion in ejector stream (JM2) if SPEC(7,N) > 0. (See Mixer usage notes. Default is 0.33).
SPEC(13,N)	If SPEC(4,N) ≤ 0 Momentum coefficient for JM1 stream, otherwise blank.
SPEC(14,N)	If SPEC(4,N) ≤ 0 Momentum coefficient for JM2 stream, otherwise blank.

SPEC(15,N)      Leave blank.

#### OUTPUT

DATOUT(1,N)    Main stream flow area, (JM1 stream), in<sup>2</sup>.  
DATOUT(2,N)    Secondary stream flow area, (JM2 stream), in<sup>2</sup>.  
DATOUT(3,N)    Total to static pressure ratio of JM1 stream.  
DATOUT(4,N)    Total to static pressure ratio of JM2 stream.  
DATOUT(5,N)    Velocity of JM1 stream, ft/s.  
DATOUT(6,N)    Velocity of JM2 stream, ft/s.  
DATOUT(7,N)    Exit mixed flow velocity, ft/s.  
DATOUT(8,N)    Static pressure difference between the JM1 and JM2 streams divided by the average static pressure.

DATOUT(9,N)    Exit area/total inlet area.

#### Mixer usage notes:

A new, improved mixer routine has been put into NNEP89. The current mixer solves the equations of mass, momentum, and energy. The current mixer will increase the exit area, if necessary, to have a valid solution to the mixing equations. Mathematically, the mixer inlet streams do not have to be at the same static pressure to have a valid mixed solution unlike the original NNEP mixer subroutine. The original mixer subroutine assumed a constant static pressure for all mixer entrance and exit streams, unless the ejector option was used.

The original velocity coefficient term has been replaced with more realistic momentum coefficient terms. These terms are used in the conservation of momentum equation. A velocity coefficient term is only applied to a flow stream's velocity contribution to momentum. The current momentum terms are applied to a flow stream's velocity and pressure contribution.

The current mixer also includes the exit static pressure and Mach number in the station property printout for all gas streams. If the current mixer has to increase the mixer exit area to get a valid solution to the mixing equations, a warning message will be included in the output.

If SPEC(7,N) > 0, the mixer will do ejector calculations. The ejector primary (JM2) throat area will be determined from the design point conditions. The JM2 stream's Mach number is input at design point and determines its exit-to-throat area ratio (like a C-D nozzle). Off design, the throat area and area ratio, with its inlet total conditions will determine the ejector primary's mass flow and inlet static conditions. Any time the ejector primary static pressure (JM2) is less than SPEC(12,N)\*ejector secondary static pressure (JM1), the mixer subroutine assumes that the ejector primary flow (JM2) separates. A message warning of this condition will be included in the output. The subroutine then recalculates the ejector primary stream's static conditions such that its static pressure is equal to the ejector secondary stream's (JM1) static pressure. The ejector primary velocity and flow area consistent with its new static pressure will be used for the remainder of that ejector calculation.

## 10.2.10 Nozzle

### INPUT

- KONFIG(1,N) 'NOZZ' or 4HNOZZ or 9
- KONFIG(2,N) Main upstream flow station number (JM1).
- KONFIG(3,N) 0
- KONFIG(4,N) Main downstream flow station number (JP1).
- KONFIG(5,N) 0
- 
- SPEC(1,N) Flow area, in<sup>2</sup>; exit for convergent, throat for C-D nozzles. The flow area is calculated at the design point based on present cycle conditions.
- SPEC(2,N) Flow or discharge coefficient ( $C_D$ ) or table reference number.  
Table =  $F(x)$  where  $x$  = nozzle overall pressure ratio.
- SPEC(3,N) Nozzle exit static pressure, psia. (Default is ambient pressure).
- SPEC(4,N) Ambient static pressure, psia. If 0, see SPEC(9,N).
- SPEC(5,N) Velocity coefficient ( $C_V$ ) or table reference number. Table =  $F(x,y)$   
where  $x = \frac{\text{Nozzle entrance total pressure}}{\text{Nozzle throat static pressure}}$ ,  $y = \frac{\text{Nozzle exit area}}{\text{Nozzle throat area}}$ .
- SPEC(6,N) Geometry Switch:  
=0 Convergent nozzle.  
=1 Type 1 C-D nozzle. Perfect expansion to ambient static pressure for all cases. If SPEC(3,N) > 0, then perfect expansion to SPEC(3,N).  
=2 Type 2 C-D nozzle. Nozzle exit to throat area ratio is calculated at design. That calculated area ratio is then used for subsequent off-design cases (unless redefined by changing SPEC(8,N)).  
=3 Type 3 C-D nozzle. Nozzle exit to throat area ratio is supplied by user (see SPEC(8,N)). If this area ratio is less than unity errors will occur.
- SPEC(7,N) Nozzle variable area switch:  
=0 Fix the area to the input value specified by SPEC(1,N) at off-design.  
=1 Vary the throat area for C-D nozzles or exit area for convergent nozzles to match the area required to pass all the flow entering the nozzle.
- SPEC(8,N) Nozzle exit to throat area ratio. Used at design point and off-design for Type 3 nozzles. Also calculated for Types 1 and 2 nozzles at design point, but only Type 2 nozzles will use this for off-design nozzle calculations. It will be ignored for Type 1 nozzles.
- SPEC(9,N) If SPEC(4,N) = 0, set SPEC(9,N) to the component number of the inlet. If SPEC(4,N) and SPEC(9,N) are less than or equal to 0, the program will use SPEC(3,1) (the free stream static pressure of inlet 1) for the nozzle ambient static pressure.
- SPEC(10,N) C-D (nozzle Types 1, 2, or 3) exit divergence half angle for divergence loss calculations (deg). This angle may be the exit angle of the shroud, the plug (if any), or both the shroud and plug, depending on the nozzle type switch for divergence loss calculations (see SPEC(12,N)). Blank if no divergence loss calculation is desired.
- SPEC(11,N) C-D nozzle axial divergent length for divergence loss calculations, in. When supplied, the divergence half angle (SPEC(10,N)) is calculated from the known throat and exit areas. This angle will take precedence over any angle specified in SPEC(10,N). Blank if no divergence loss calculation is desired.

SPEC(12,N) Geometry (nozzle type) switch for C-D nozzle divergence loss calculations:  
 =0 (default) disables divergence calculations, divergence loss coefficient set to unity.  
 =1 Axisymmetric C-D nozzle.  
 =2 2-D exit C-D nozzle.  
 =3 Axisymmetric plug C-D nozzle with equal wall exit half angles on plug and shroud.  
 =4 2-D exit plug C-D nozzle with equal wall exit half angles on plug and shroud.  
 =5 Axisymmetric plug C-D nozzle with cylindrical (non-divergent) shroud.  
 =6 2-D exit plug (wedge) C-D nozzle with parallel (non-divergent) shroud walls.

SPEC(13,N) Blank

SPEC(14,N) Blank

SPEC(15,N) Blank

#### OUTPUT

DATOUT(1,N) Gross jet thrust, lb.

DATOUT(2,N) Actual jet velocity, ft/s.

DATOUT(3,N) Total to static pressure ratio at throat.

DATOUT(4,N) Nozzle exit area, in<sup>2</sup>.

DATOUT(5,N) Nozzle throat area, in<sup>2</sup>.

DATOUT(6,N) Flow or discharge coefficient,  $C_D$ .

DATOUT(7,N) Velocity coefficient,  $C_v$ .

DATOUT(8,N) Critical pressure ratio at throat.

DATOUT(9,N) Overall nozzle pressure ratio: nozzle entrance total to exit static.

#### Nozzle usage notes:

The only method that the original version of NNEP used for converging-diverging nozzle analysis was to "float" or "rubberize" the nozzle exit area such that the flow was fully expanded to ambient static conditions. This method yields the maximum thrust, but it also implies a variable geometry nozzle. The current nozzle routine is modified to handle fixed area ratio C-D nozzles. When the exit to throat area ratio is fixed, the nozzle will yield maximum thrust only at its design pressure ratio. If a fixed area ratio nozzle is operating at a pressure ratio less than or greater than its design point pressure ratio, the nozzle is said to be overexpanded or underexpanded, respectively. Due to the unequal exit static and ambient pressures, a pressure thrust or a pressure drag term must be included in the gross thrust calculations. No separation criteria are applied for overexpanded nozzles; hence, the pressure drag term applied may be unusually large for these cases.

Another new option for the nozzle component is SPEC(3,N) - the nozzle exit static pressure. The user can now specify this pressure. The nozzle will expand the nozzle flow to the desired static pressure, regardless of whether the flow is overexpanded or underexpanded. Note that losses due to expansion to other than ambient conditions are categorized as thermodynamic losses. As always, any losses due to viscous effects must be accounted for by furnishing velocity

coefficients and/or discharge coefficients. These loss coefficients are only applied to the momentum thrust calculations.

The fixed area ratio calculations are invoked only when certain inputs are specified. Thus input files written for earlier versions of NNEP will produce the same results as before.

The two methods for fixing the nozzle exit to throat area ratio are:

1. Set SPEC(6,N) = 2 (a "Type 2" C-D nozzle). For Type 2 C-D nozzles, the nozzle exit to throat area ratio required for perfect expansion is calculated at the design point. That area ratio is stored in SPEC(8,N) and is used for all subsequent off-design cases. Unless, of course, the user specifies a control to vary the area ratio or inputs a new area ratio in SPEC(8,N).  
Note: the user, while exercising caution, may at any time "redesign" a Type 2 C-D nozzle by setting IDONE(N) = 0 or by setting SPEC(6,N) = 1 for that case and then resetting SPEC(6,N) = 2.
2. Set SPEC(6,N) = 3 (a "Type 3" C-D nozzle). For Type 3 C-D nozzles, the user specifies the desired area ratio by supplying a value in or putting a control on DATINP(8,N). The area ratio in DATINP(8,N) is used exclusively in all nozzle thrust calculations.

Another addition to the nozzle routine is the ability to apply divergence (i.e., non-axial) gross thrust losses to converging-diverging nozzles. This is discussed further in reference 6. These losses are accounted for by applying an analytically derived, geometry dependent divergence loss coefficient to the gross momentum thrust. The user is offered a choice of six different C-D nozzle geometries (shown in Table 7) for divergence loss calculations.

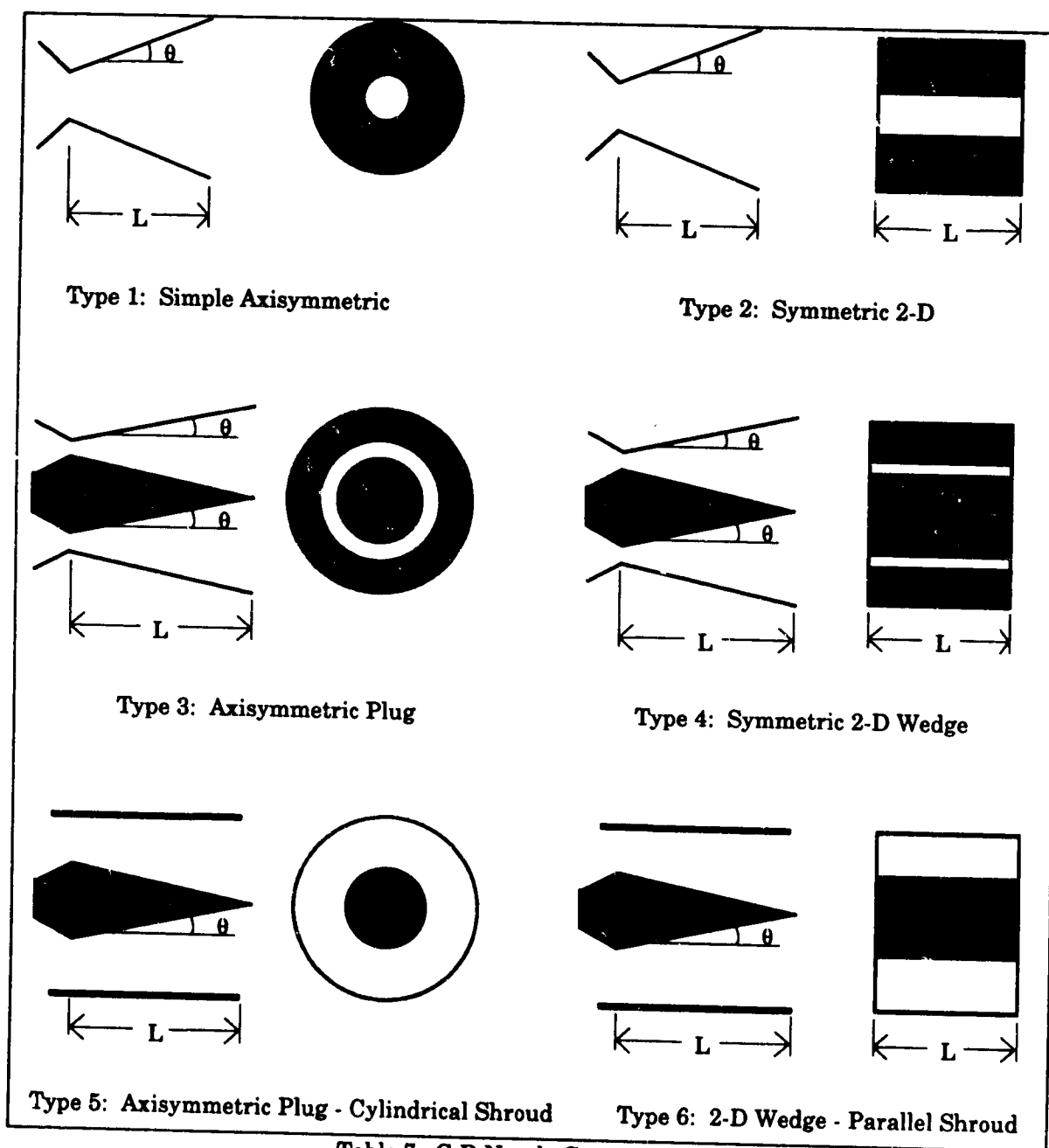


Table 7. C-D Nozzle Geometry Types.

### 10.2.11 Load or Propeller

#### INPUT

KONFIG(1,N) 'LOAD' or 4HLOAD or 10  
KONFIG(2,N) 0  
KONFIG(3,N) 0  
KONFIG(4,N) 0  
KONFIG5,N) 0

SPEC(1,N) Horsepower (negative for load or positive for power input to cycle) or table reference number. Table = F(x) where x = actual shaft rotational speed, rpm.

SPEC(2,N) Propeller efficiency ( $\frac{\text{Thrust} \cdot \text{Flight Velocity}}{\text{Shaft power input}}$ ) or table reference number. Note: loads other than propellers may be connected to shafts; for such loads only SPEC(1,N) is specified, all others must be blank. Table = F(x,y,z) where x = coefficient of power,  $\frac{550 \text{ SHP}}{\rho n^3 D^5}$ , y = advance ratio,  $\frac{\pi \cdot \text{Flight velocity}}{\text{Tip speed}}$ , z = free stream Mach number.

SPEC(3,N) Propeller thrust divided by shaft horsepower for static thrust calculations when tables are not used, lb/HP.

SPEC(4,N) Design point power loading (shaft horsepower divided by propeller diameter squared).

SPEC(5,N) Design tip speed of the propeller, ft/s.

SPEC(6,N) Blank

SPEC(7,N) Blank

SPEC(8,N) A constant scaling factor which is multiplied by the results taken from the propeller map. This input scales the efficiency, thrust, and coefficient of thrust when propeller maps are specified in SPEC(2,N). SPEC(8,N) will override SPEC(9,N) if both are input non-zero.

SPEC(9,N) Desired propeller design point efficiency when using propeller maps. This input determines a scale factor which will be applied to all subsequent off-design cases. SPEC(8,N) will override SPEC(9,N) if both are input non-zero.

SPEC(10,N)-  
through-  
SPEC(15,N) Blank

#### OUTPUT

DATOUT(1,N) Load power, hp.

DATOUT(2,N) Actual shaft rotational speed, rpm.

DATOUT(3,N) Propeller thrust ( $\frac{\text{Shaft power input} \cdot \eta}{\text{Flight velocity}}$ ), lb. (If flight velocity is zero, the equation for propeller thrust becomes indeterminate and will be set to zero if SPEC(3,N) is not input or tables are not used.)

DATOUT(4,N) Advance ratio,  $\frac{\pi \cdot \text{Flight velocity}}{\text{Tip speed}}$ .

DATOUT(5,N) Coefficient of power,  $\frac{550 \text{ SHP}}{\rho n^3 D^5}$ .

DATOUT(6,N) Coefficient of thrust,  $\frac{\text{Thrust}}{\rho n^2 D^4}$ .

DATOUT(7,N) Propeller tip speed, ft/s.

DATOUT(8,N) Propeller diameter, ft.

DATOUT(9,N) Adjusted propeller efficiency (using SPEC(8,N), or SPEC(9,N)).



### 10.2.12 Shaft

#### INPUT

KONFIG(1,N) 'SHFT' or 4HSHFT or 11  
KONFIG(2,N) Component number (JM1) of the first component coupled to this shaft.  
KONFIG(3,N) Component number (JM2) of the second component coupled to this shaft.  
KONFIG(4,N) Component number (JP1) of the third component coupled to this shaft.  
KONFIG(5,N) Component number (JP2) of the fourth component coupled to this shaft.

SPEC(1,N) Actual shaft rotational speed, rpm.  
SPEC(2,N) Gear ratio of JM1 component, component rpm/shaft rpm.  
SPEC(3,N) Gear ratio of JM2 component, component rpm/shaft rpm.  
SPEC(4,N) Gear ratio of JP1 component, component rpm/shaft rpm.  
SPEC(5,N) Gear ratio of JP2 component, component rpm/shaft rpm.  
SPEC(6,N) Mechanical efficiency of JM1 component, actual power/ideal power.  
SPEC(7,N) Mechanical efficiency of JM2 component, actual power/ideal power.  
SPEC(8,N) Mechanical efficiency of JP1 component, actual power/ideal power.  
SPEC(9,N) Mechanical efficiency of JP2 component, actual power/ideal power.  
SPEC(10,N)-  
through-  
SPEC(15,N) Blank

#### OUTPUT

DATOUT(1,N) Net shaft power, the sum of DATOUT(1,N) of all components connected to the shaft, hp.  
DATOUT(2,N) Actual shaft rotational speed, rpm.  
DATOUT(3,N) Actual rotational speed of component JM1, rpm.  
DATOUT(4,N) Actual rotational speed of component JM2, rpm.  
DATOUT(5,N) Actual rotational speed of component JP1, rpm.  
DATOUT(6,N) Actual rotational speed of component JP2, rpm.  
DATOUT(7,N) Blank  
DATOUT(8,N) Ratio of net shaft power to total power. Total power is the sum of the absolute values of all loads on this shaft divided by two.  
DATOUT(9,N) Blank

#### Shaft usage notes:

A variable control is required for off design operation of the program to drive the shaft work error ( DATOUT(8,N) ) to zero. If one shaft is to be connected to another shaft in order to have more than 4 components on the same shaft, the user must configure the shafts using the following rules. The lower component number shaft must be the first component (JM1) of the higher number shaft. At least one turbine must be on the higher number shaft. The control on horsepower balance must vary the shaft speed of the lower number shaft to drive the work error of the higher number shaft to zero.

### 10.2.13 Variable Control

Since NNEP89 does not have any preset engine configurations, the user define as input to the program any flow or work errors that the program must drive to zero. The user may also specify that some components operate at a designated condition (such as operating compressors at a constant surge margin). Variable controls specify to the program what independent variables to change to get the engine model to operate in the desired manner.

#### INPUT

KONFIG(1,N) 'CNTL' or 4HCNTL or 12  
KONFIG(2,N) 0  
KONFIG(3,N) 0  
KONFIG(4,N) 0  
KONFIG(5,N) 0

Note: Inputs for controls are different from those of other components. The SPCNTL variable is used for input at the design point, but the SPEC variable must be used to read in values for off-design cases. SPCNTL values do not retain their position when converted to SPEC array values.

The first two inputs describe the independent variable.

SPCNTL(1,N) The SPEC number of the independent variable.

SPCNTL(2,N) The component number containing the independent variable being varied

The next four inputs describe the dependent variable.

SPCNTL(3,N) Type of dependent variable being specified.  
= 'STAP', 4HSTAP, or 100 if the dependent variable is a station property.  
= 'DOUT', 4HDOUT, or 200 if the dependent variable is a DATOUT.  
= 'PERF', 4HPERF, or 400 if the dependent variable is a performance property.

SPCNTL(4,N) Number that specifies the desired station property, DATOUT, or performance property.

SPCNTL(5,N) =Flow station containing the dependent variable if it is a station property.  
=Component number containing the dependent variable if it is a DATOUT.  
=0 if the dependent variable is a performance property.

SPCNTL(6,N) Value of the dependent variable to be achieved.

SPCNTL(7,N) Tolerance of control - input as a fraction of the desired value.  
= 0 Control is turned off.  
= 1 Default value of 0.001 is used (0.0001 if NVOPT  $\neq$  0).

SPCNTL(8,N) Minimum allowable value of independent variable - (if 0, this input is ignored).

SPCNTL(9,N) Maximum allowable value of independent variable - (if 0, this input is ignored).

The next three inputs are used to define a variable target value which can be used in place of SPCNTL(6,N). Otherwise they must be left blank.

SPCNTL(10,N) Type of variable being specified as a target value.  
= 'STAP' or 4HSTAP, or 100 - if the target value is a station property.  
= 'DOUT' or 4HDOUT, or 200 - if the target value is a DATOUT.

- = 'DINP' or 4HDINP, or 300 - if the target value is a SPEC.
- = 'PERF' or 4HPERF, or 400 - if the target value is a performance property.
- SPCNTL(11,N) Number that specifies the station property, DATOUT, SPEC or performance property of SPCNTL(10,N).
- SPCNTL(12,N) = Flow station number if target value is a station property.
- = Component number if the target value is a DATOUT or SPEC.
- = 0 if the target value is a performance property.

### OFF-DESIGN SPECs

For off-design, the control specifications can only be modified using the SPEC array. This array is defined as follows:

- SPEC(1,N) Fraction of target value used for marching (see Marching explanation below).
- SPEC(2,N) Minimum allowable value. See variable control usage notes.
- SPEC(3,N) Maximum allowable value. See variable control usage notes.
- SPEC(4,N) The SPEC number of the independent variable.
- SPEC(5,N) Value of the dependent variable to be achieved,  
(Only if SPCNTL(10,N) at design point or SPEC(10,N) at off-design equals 0).
- SPEC(6,N) Station property number if dependent variable is a station property, otherwise blank.
- SPEC(7,N) DOUT number if dependent variable is a DATOUT, otherwise blank.
- SPEC(8,N) Performance property number if dependent variable is a 'PERF', otherwise blank.
- SPEC(9,N) Tolerance on value to be achieved. If zero, control is off. If SPEC(9,N) = 1 default value of 0.001 is used (0.0001 if NVOPT ≠ 0).
- SPEC(10,N) through SPEC(12,N) have the same value as SPCNTL(10,N) through SPCNTL(12,N)
- SPEC(10,N) Type of dependent variable, (STAP, DOUT, DINP or PERF).
- SPEC(11,N) Number that specifies the station property, DATOUT, SPEC or performance property for a variable target value.
- SPEC(12,N) Flow station number or component number for the variable target value.
- SPEC(13,N) Blank
- SPEC(14,N) Blank
- SPEC(15,N) Blank

### OUTPUT

There are no DATOUTS or output for CNTLs.

Variable control usage notes:

Unless the user is using optimization variables, the input maximum and minimum values will not have any affect on the value of the independent variable used by the program. If the program exceeds a limit that the user input, the program will simply write a warning message in the output file. If the user is using optimization and the program exceeds a limit, the program will analyze the optimization function for exceeding the limits. That penalty may prevent the program from exceeding the limits.

The array of available station properties are:

<u>Station Property Number</u>	<u>Definition</u>
1	Weight flow, lb/s
2	Total pressure, psia
3	Total temperature,
4	Fuel-to-air ratio
5	Referred flow, $w\sqrt{T}/P$ Note: corrected flow, $w\sqrt{\theta}/\delta$ , is printed on the output $w\sqrt{T}/P = 1.5497 w\sqrt{\theta}/\delta$
6	Mach number
7	Static pressure, psia
8	Interface corrected flow error = $\frac{W_{corr1} - W_{corr2}}{\frac{1}{2}(W_{corr1} + W_{corr2})}$

where  $W_{corr1}$  is the corrected flow out of the component immediately upstream of the flow station and

$W_{corr2}$  is the corrected flow out of the component immediately downstream of the flow station.

The array of available performance properties are:

<u>Performance Property Number</u>	<u>Definition</u>
1	Total engine airflow (lb/s)
2	Gross jet thrust (lb)
3	Fuel flow (lb/hr) (includes oxidizer flow from gas generator)
4	Net jet thrust (lb)
5	$TSFC = \frac{\text{Fuel flow (lb/hr)}}{\text{Net jet thrust (lb)}}, \text{ lb/lb-hr}$
6	Net thrust/airflow, lb-s/lb
7	Total inlet drag, lb
8	Total brake shaft horsepower
9	Net thrust with installation drag, lb
10	Net TSFC with installation drag, lb/lb-hr
11	Inlet drag (lip, bleed and spillage), lb
12	Boattail drag, lb
13	Blank
14	Blank
15	Sum of SPEC(2,N) of all turbines - one. See turbine cooling bleed calculations for usage.
16	Emission Index, grams $NO_x$ per kilogram fuel
17	Total shaft horsepower of all propellers

### 10.2.13.1 Variable Target Values

If SPCNTL(10,N) is not equal to zero, then a variable target value will be used instead of SPCNTL(6,N). SPCNTL(10,N), SPCNTL(11,N), and SPCNTL(12,N) are used to allow the user to have the program drive the dependent variable equal to the value or the multiple of a specified station property, data output, data input (SPEC), or performance property.

The following example demonstrates one use of variable target values. Suppose one has a turbofan with 2 nozzles and would like them to have equal jet velocities. The main nozzle is component 8, the secondary nozzle is component 20. Just before the secondary nozzle is duct 19, which is used as an afterburner. One can now vary the temperature (SPEC 4) of the afterburner (DUCT 19) to get the jet velocity (DATOUT 2) of nozzle 20 equal to the jet velocity of nozzle 8. This must be done in this manner because we do not know "a priori" the jet velocity of nozzle 8. For this example, the control will be number 41 and will have a tolerance of 0.002. The minimum and maximum values for the burner temperature are 800 and 3600 °R, respectively. The control would be input as follows:

```
KONFIG(1,41)='CNTL',SPCNTL(1,41)=4,19,'DOUT',2,20,0,0.002,800,3600,4HDOUT,2,8,
```

### 10.2.13.2 Marching

Marching is a special option on variable controls which allows the user to reset the target value ( SPEC(5,N) ) of the variable control to a multiple of that value from the previous, converged case. Marching may be initiated in any off-design case by setting SPEC(1,N) of the appropriate variable control to the desired multiplying factor that is to be used for the marching. As long as SPEC(1,N) of the variable control is not zero, the multiplying factor will be applied in all subsequent cases. In each case the value of the dependent variable from the previous, converged case is multiplied by the SPEC(1,N) value and this new value is used as the target value of the variable control.

If a variable target value is used, marching will work differently than described above. In this case the variable specified as the target (SPCNTL(10,N) through SPCNTL(12,N)) is multiplied by SPEC(1,N). As the program iterates to a converged solution, this result is continuously updated to become the new target value of the variable control.

The following example demonstrates the use of marching. Suppose one wishes to make a plot of thrust over airflow versus nozzle area at Mach 1.4 and 40000 feet. A variable control component is included defining nozzle area as the independent variable and thrust to airflow ratio as the dependent variable. An actual target value is not input because it will be computed by the program. This control is turned off for the design case. Assuming the component number of the nozzle is 10, the variable control would be specified as follows:

```
KONFIG(1,20)='CNTL', SPCNTL(1,20)=1,10,'PERF',6,0,0,0
```

Now the engine design point is executed as well as any desired off-design cases. At the point where the user wishes to begin marching, the variable control is turned on and the marching value is set to its desired value (1.1 in this example) as follows:

```
&D SPEC(1,20)=1.1,SPEC(9,20)=1, &END
```

This may be followed by as many subsequent cases as desired. When the program detects SPEC(1,20) is not equal to zero, it will take the last value of the dependent variable (in this case PERF(6)), multiply it by SPEC(1,N), and store that value in SPEC(5) of the variable control. The control will then drive the dependent variable to this new target value.

Marching can also be used when there is a variable target value. Using the previous example of the variable target value from Section 10.2.13.1, suppose that the user wants the jet velocity of the secondary nozzle to be 1.35 times the velocity of the primary nozzle. The user would set SPEC(1,41)=1.35 and the program would vary the burner temperature of duct 19 to get the jet velocity (DATOUT 2) of nozzle 20 to 1.35 times the jet velocity (DATOUT 2) of nozzle 8.

#### 10.2.14 Variable Optimization

##### **INPUT**

KONFIG(1,N)	'OPTV' or 4HOPTV or 13
KONFIG(2,N)	0
KONFIG(3,N)	0
KONFIG(4,N)	Component number containing the independent variable.
KONFIG(5,N)	0
SPEC(1,N)	Blank
SPEC(2,N)	Minimum allowable value of the optimized dependent variable (if = 0, there is no minimum constraint).
SPEC(3,N)	Maximum allowable value of the optimized dependent variable (if = 0, there is no maximum constraint).
SPEC(4,N)	A value from 1 to 15 indicating which SPEC of component KONFIG(4,N) is the independent variable.
SPEC(5,N)- through-	
SPEC(8,N)	Blank
SPEC(9,N)	Switch to turn the OPTV variable on or off. If set = 0, the variable is off. If set = 1, the variable is on.
SPEC(10,N)- through-	
SPEC(15,N)	Blank

There are additional inputs to NNEP89 when OPTV components are present. OPTVs tell the program which variables are the independent variables. The next variables tell the program whether or not to execute with optimization and which variable is the "cost function" that is being minimized or maximized (the dependent variable). These variables are:

TOLOPT	Criteria of convergence on the dependent variable. The default is 0.0002.
NJOPT	Component number which indicates the location of the dependent variable (if zero, the dependent variable is a performance property, not a DATOUT parameter).
NVOPT	To turn off the optimization, NVOPT must be set to zero or all OPTVs must be turned off. If NVOPT is positive, the optimum is a minimum; if NVOPT is negative, the optimum is a maximum. If NJOPT equals zero: A value of 1 to 17 indicating which performance property is the dependent variable (see list of available performance properties in description of CNTL). If NJOPT not equal to zero: A value of 1 to 9 indicating which DATOUT of component number NJOPT is the dependent variable.

## OUTPUT

There is no DATOUT output array for OPTV components.

### Optimization usage notes:

The optimal variable component (OPTV) allows the user to maximize or minimize a "cost function" (i.e., the performance property or component output variable being optimized). An OPTV that is turned on (SPEC(9,N)=1) tells the program what variable (or variables if the user has multiple OPTV's that are turned on) the program can vary to minimize or maximize a "cost function". OPTV's are only used when optimization has been turned on (using variable NVOPT). Therefore optimization only takes place if NVOPT  $\neq$  0, and at least one OPTV is turned on.

As an example of the use of an OPTV, let us assume that we have marched to Mach 1.4 at 40000 feet and then throttled back to 50 percent F/Wa (See the previous example on marching in the description of CNTLs). We can now set SPEC(1,20)=1 to hold the F/Wa at the present value. If we want to minimize the TSFC while holding F/Wa constant and varying the turbine inlet temperature, we would do the following:

Assume that component 5 is the main burner. To perform the desired optimization a new component would be input as follows:

KONFIG(1,21)='OPTV',0,0,5,0,SPEC(1,21)=0,0,0,4,0,0,0,0,0,

This input record says that DATINP(4) (the burner outlet temperature) of component 5 (the main burner) is the independent variable. There is no minimum or maximum value and since SPEC(9,21)=0, it is off.

Now to activate the optimizer we set SPEC(9,21)=1 and NVOPT=5 to minimize the TSFC.

Debugging: The variable DEBUG can be used to get intermediate output during optimization. If DEBUG=1, the program will print to the output dataset the following values:

- 1)  $\frac{\text{New value of the cost function}}{\text{Original value of the cost function}}$
- 2)  $\frac{\text{New value of the penalty function}}{\text{Original value of the penalty function}}$   
( =1, if no variables have exceeded their user-defined limits),
- 3)  $\frac{\text{New value of best value of the cost function}}{\text{Original value of the cost function}}$
- 4)  $\frac{\text{New values of the optimization variables}}{\text{Original values of the optimization variables}}$



### 10.2.15 Variable Limit

#### INPUTS

The inputs at the design point are

KONFIG(1,N) 'LIMV' or 4HLIMV or 14  
KONFIG(2,N) 0  
KONFIG(3,N) 0  
KONFIG(4,N) 0  
KONFIG(5,N) 0

SPEC(1,N) Blank  
SPEC(2,N) Minimum allowable value.  
SPEC(3,N) Maximum allowable value.  
SPEC(4,N) Type of limit variable being specified.  
= 'STAP', 4HSTAP, or 100 if the limit variable is a station property.  
= 'DOUT', 4HDOUT, or 200 if the limit variable is a DATOUT.  
= 'PERF', 4HPERF, or 400 if the limit variable is a performance property.  
SPEC(5,N) =Flow station containing the limit variable if it is a station property.  
=Component number containing the limit variable if it is a DATOUT.  
=0 if the limit variable is a performance property.  
SPEC(6,N) Component number, flow station number, or blank.  
SPEC(7,N) Blank  
SPEC(8,N) Blank  
SPEC(9,N) On-off switch, 1 = On, 0 = Off.

#### OUTPUT

There is no DATOUT output array for LIMV components.

Variable limit usage notes:

When a limit defined by the LIMV variable is exceeded, a warning message will be printed on the output file. If optimization is in effect, the criteria of merit, or the "cost function", will be penalized by a penalty function to drive the user away from exceeding the minimum and maximum limits.

### 10.2.16 Variable Scheduling

#### INPUT

KONFIG(1,N) 'SKED' or 4HSKED or 15  
KONFIG(2,N) Component number containing the dependent (scheduled) variable.  
KONFIG(3,N) Flow station number or component number containing the first independent variable on the map. X  
KONFIG(4,N) Flow station number or component number containing the second independent variable on the map. Y  
KONFIG(5,N) Flow station number or component number containing the third independent variable on the map. Z

SPEC(1,N) =0 turn schedule off,  
=1 turn schedule on.  
SPEC(2,N) SPEC number of the dependent variable of component KONFIG(2,N).  
SPEC(3,N) Scale factor on dependent variable (usually input as 1).  
SPEC(4,N) Table reference number containing the schedule map.  
SPEC(5,N) Location of first independent variable.  
1-8 = STAP(1) - STAP(8) of flow station X (KONFIG(3,N)).  
11-19 = DATOUT(1,N) - DATOUT(9,N) of component X (KONFIG(3,N)).  
21-35 = SPEC(1,N) - SPEC(15,N) of component X (KONFIG(3,N)).  
SPEC(6,N) Scale factor on SPEC(5,N) (usually input as 1).  
SPEC(7,N) Location of second independent variable.  
1-8 = STAP(1) - STAP(8) of flow station Y (KONFIG(4,N)).  
11-19 = DATOUT(1,N) - DATOUT(9,N) of component Y (KONFIG(4,N)).  
21-35 = SPEC(1,N) - SPEC(15,N) of component Y (KONFIG(4,N)).  
SPEC(8,N) Scale factor on SPEC(7,N) (usually input as 1).  
SPEC(9,N) Location of 3rd independent variable.  
1-8 = STAP(1) - STAP(8) of flow station Z (KONFIG(5,N)).  
11-19 = DATOUT(1,N) - DATOUT(9,N) of component Z (KONFIG(5,N)).  
21-35 = SPEC(1,N) - SPEC(15,N) of component Z (KONFIG(5,N)).  
SPEC(10,N) Scale factor on SPEC(9,N) (usually input as 1).  
SPEC(11,N)-  
through-  
SPEC(15,N) Blank

#### OUTPUT

There are no DATOUTs for SKED components

**Variable scheduling usage notes:**

Many components have SPECs which may be specified by a map that is a function of up to three variables (value =  $F(x,y,z)$ ). The SKED component allows the user to designate any SPEC of the first 11 types of component (i.e. the engine components) to be a function of up to three independent variables. The independent variables are specified by the user as indicated below. The value of the SPEC is determined by interpolating a user supplied map. The map format is the same as for other map tables.

As an example as how to use a SKED, let us assume that we want the program to set the temperature (SPEC 4) of the burner (DUCT 11) as a function of:

the compressor pressure ratio (DATOUT 9) of compressor 3 ,

the mass flow rate (station property 1, see Table 3) entering burner 11 at flow station number 22 , and

the rpm (SPEC 1) of shaft 31.

We have made a table, in the correct performance map format, that has burner temperature as a function of compressor pressure ratio, burner entrance mass flow, and shaft rpm. The table reference number is 9001 and the SKED will be component 39.

All scale factors are 1.0 and the schedule is turned on. The schedule would be input as follows:

KONFIG(1,39)='SKED',11,3,22,31,SPEC(1,39)=1,4,1.0,9001,19,1.0,3,1.0,21,1.0,

### 10.2.17 Conditional Control

#### INPUT

KONFIG(1,N) VCNT or 4HVCNT or 16  
KONFIG(2,N) 0  
KONFIG(3,N) 0  
KONFIG(4,N) 0  
KONFIG(5,N) 0

SPEC(1,N) New value for the SPEC variable that conditional controls will reset if triggered. If a conditional control is being used to turn a variable control on or off, a value of SPEC(1,N) less than or equal to zero will turn the variable control off. A value of SPEC(1,N) greater than zero will turn the variable control on. Note: if the tolerance of a variable control is input as 1, the program will use the default of 0.001 (or 0.0001 if NVOPT  $\neq$  1).

SPEC(2,N) Component number for the variable that the conditional control will reset.

The next three SPECs are used to set up a variable that will trigger the conditional control. The value of this variable is monitored in relation to the trigger value to determine if the conditional control will reset the desired SPEC variable. This variable is referred to as the trigger variable, and is not to be confused with the SPEC variable that the conditional control will reset.

SPEC(3,N) The type of trigger variable.  
'STAP', 4HSTAP, or 100 if trigger variable is a station property.  
'DOUT', 4HDOUT, or 200 if trigger variable is a data output (DATOUT).  
'DINP', 4HDINP, or 300 if trigger variable is a data input (SPEC).  
'PERF', 4HPERF, or 400 if trigger variable is a performance property.

SPEC(4,N) Station property number of trigger variable if SPEC(3,N)='STAP'.  
Data output number of trigger variable if SPEC(3,N)='DOUT'.  
Data input SPEC number of trigger variable if SPEC(3,N)='DINP'.  
Performance property number of trigger variable if SPEC(3,N)='PERF'.

SPEC(5,N) Flow station number of trigger variable if SPEC(3,N)='STAP'.  
Component number of trigger variable if SPEC(3,N)='DOUT'.  
Component number of trigger variable if SPEC(3,N)='DINP'.  
0 if SPEC(3,N)='PERF'.

SPEC(6,N) Trigger value, the limit that has to be exceeded (either greater than or less than) to make the conditional control reset SPEC variable SPEC(12) of component SPEC(2).

SPEC(7,N) This SPEC is only necessary if the user defines more than one conditional control to reset one particular variable.  
=0 This conditional control will not reset the desired SPEC variable (SPEC(12,N) of component SPEC(2,N)) more than once per case. This prevents multiple conditional controls from resetting one SPEC variable back and forth during the same case.  
=1 The conditional control will reset the desired SPEC variable every time the target variable has been exceeded.

SPEC(8,N)	=0 Do not reset SPEC(12,N) of component SPEC(2,N) at end of case. =1 Reset SPEC(12,N) of component SPEC(2,N) to the value it had at the beginning of the case before processing the next case.
SPEC(9,N)	=0 Conditional control is nonactive. =1 Set SPEC(12,N) of component SPEC(2,N) to SPEC(1,N) if triggering variable is <u>greater than</u> the triggering value, SPEC(6,N). =-1 Set SPEC(12,N) of component SPEC(2,N) to SPEC(1,N) if triggering variable is <u>less than</u> the triggering value, SPEC(6,N).
SPEC(10,N)	Blank
SPEC(11,N)	This input variable is only used if the conditional control is resetting SPEC(9,N) of a variable control. =0 Do not reset the independent value of the variable control. =1 Reset the independent value of the variable control to the value it had at the beginning of the case before processing the next case.
SPEC(12,N)	Number of the SPEC for component SPEC(2,N) that the conditional control will reset. Default is 9.
SPEC(13,N)	Blank
SPEC(14,N)	Blank (If the conditional control is used to reset SPEC(9,N) of a variable control, this location is used to store the original value of the independent variable of the variable control.)
SPEC(15,N)	Blank (used to store the original value of the SPEC variable that the conditional control resets, before any conditional controls are activated).

## OUTPUT

There are no DATOUTs for VCNTs.

Conditional control usage notes:

Conditional controls (VCNT) allow the users to define variables that the program will monitor. If a variable exceeds a preset, user-defined limit, the conditional control will reset another user-defined variable. This user-defined variable could be one variable input (SPEC) for any component, such as: 1) resetting burner temperature or bypass ratios; 2) turning variable schedules on or off; or 3) turning variable controls on or off.

The uses for conditional controls are best explained by examples. For the first use of conditional controls, suppose the user wants to turn on burner augmentation at Mach numbers above 1.1, but does not want to input the variables to do that directly. The user just sets up a conditional control to reset the augmentor temperature to the desired level (turn on augmentation) if the Mach number is greater than 1.1 and another conditional control to reset the augmentor temperature to zero (turn off augmentation), if the Mach number is less than 1.1. For a second example, which involves the second and third uses of conditional controls, suppose that a user has an inlet airflow schedule to set engine airflow versus Mach number. The user is throttling the engine thrust back at various Mach numbers, but can only throttle the engine so far at a constant airflow. Then the airflow must be allowed to be changed or the cycle will not converge. The user would set up conditional controls to watch one cycle parameter, such as compressor surge margin, or burner temperature to turn off the airflow schedule (SPEC(1,N) of the correct variable schedule component) and maybe turn on a variable control to vary airflow to meet the desired engine performance level.

One conditional control can only be used to reset one SPEC variable to one new value, or to turn one variable schedule or variable control either on or off. For example, to turn a variable control on at one cycle condition, but turn it back off during the same case requires 2 conditional controls. At the end of that case, conditional controls can be configured to reset all variables that they have changed. The independent variables of variable controls, activated by conditional controls can also be reset to their original values. The default option is to leave all SPEC variables and variable controls in their present states and values at the end of that case.

Conditional controls are handled in the following manner. Before the program actually starts to iterate on the solution, it checks all active conditional controls whose trigger variable is a data input (SPEC(3,N)='DINP'). If a data input has already exceeded its limit, the program will reset the specified SPEC variable and write to the output dataset that it has reset a SPEC variable. Then the program starts execution of the case. During the case, the program reaches a solution. The program will check each active conditional control to see if it has exceeded the specified limit for the specified trigger variable. If a limit has been exceeded for any conditional control, the program will reset the specified SPEC variable and write to the output dataset that it has reset a SPEC variable. The program will check all active conditional controls. Even if only one conditional control has reset a SPEC variable, the program will re-execute the case. To prevent the program from entering an infinite loop, it will only re-execute a case a maximum of 10 times with any variables, variable schedules, or variable controls changed by conditional controls. At the end of the case, if SPEC(8,N) of the conditional control is equal to 1, the conditional control will restore the original value of the SPEC variable. Otherwise the SPEC variable will stay at its current setting. If the conditional control was turning a variable control either on or off, setting SPEC(11,N) of the conditional control to 1 would cause the conditional control to reset the independent variable of the variable control to its value from the beginning of the case. Otherwise, the independent variable of the variable control will also remain at its current value.

### 10.3 TABLE DATA INPUTS FORMAT

Consider the table data as three dimensional, composed of a series of planes with each plane assigned a value called Z. Then, on each Z plane, the dependent variable (ordinate axis), is a two dimensional function of X (abscissa axis) and Y (see Figure 2). This three dimensional table is often referred to as a map.

Each table has the following input format:

- Record 1 Table Reference Number (Integer, columns 2-5).  
Table Identification Label, (Alpha-numeric, columns 6-75).
- Record 2 Z - Identifier (Alpha-numeric, columns 1-4).  
NZ - Number of Z values (Integer, columns 6-7).  
Z - Variable values, 7F10., beginning in column 11.  
If needed, extra cards follow 10X,7F10. format. Z values must be in ascending order.
- Record 3 Y - Identifier (Alpha-numeric, columns 1-4).  
NY - Number of Y values (Integer, columns 6-7).  
Y - Variable values, 7F10., beginning in column 11.  
If needed, extra cards follow 10X,7F10. format. Y values must be in ascending order.
- Record 4 X - Identifier (Alpha-numeric, columns 1-4).  
NX - Number of X values (Integer, columns 6-7).  
X - Variable values, 7F10., beginning in column 11.  
If needed, extra cards follow 10X,7F10. format. X values must be in ascending order.
- Record 5 F(X,Y,Z) - Identifier (Alpha-numeric, columns 1-4).  
NX - Number of X values (Integer, columns 6-7).  
F(X,Y,Z) - Variable values, 7F10., beginning in column 11.  
If needed, extra cards follow 10X,7F10. format.  
These values correspond to the values in the X identifier record.
- Last Record-The last record must consist of the 3 characters EOT in columns 1-3.

#### Restrictions

- 1) NZ, NY, NX may not be blank or zero.
- 2) Maximum of 70 tables.
- 3) NZ, NY, NX are limited to 100 or less.
- 4) a zero or blank table reference number will halt the read.
- 5) A storage error message will result if table storage exceeds the limit set in Subroutine TREAD.

#### 10.4 CHEMICAL EQUILIBRIUM INPUT INSTRUCTIONS

The chemical dissociation option enables the program to handle just about any combination of chemical species for which thermodynamic data is available. (The default thermodynamic routine used in the NNEP89 program is preset with properties for air and JP4 fuel.) The dissociation option uses a rewritten version of the computer program for calculation of complex Chemical Equilibrium Compositions (CEC) (ref. 16), to calculate thermodynamic properties.

To use the chemical equilibrium option, the user must set ICEC=1 in the global &D Namelist input. This will cause the program to read in the necessary CEC information that follows. The CEC input must contain information about the reactants if the equilibrium option is invoked, and may contain thermodynamic data and desired debugging options in a separate CEC Namelist input block. The thermodynamic data will be processed by the CEC program and written to a binary thermodynamics file on unit 24. If this binary dataset already exists from a previous run, the thermodynamic data is not required in the input dataset. This will also save the computational time that CEC uses sorting and processing the thermodynamic data. The format for this thermodynamic data is discussed in reference 16 and will not be discussed here. The CEC &INPT2 Namelist input can contain all the inputs listed in the original CEC program, with the addition of a few extra variables that can be used to debug either the CEC routines or the main NNEP89 routines. The changes to the CEC program, the options available for its use and debugging, and its incorporation in the NNEP89 program are discussed in reference 4. The REACTANT input is often confusing for the novice user. An input file with some of the most common REACTANT species (air, JP fuel, methane, and liquid hydrogen) is included with each release of the program. The exact format for the reactants input is listed in Table 8 and an example of the inputs necessary to execute a case with the CEC routines will be discussed later. If the binary thermodynamic dataset already exists, the only input required to use the CEC thermodynamic routines is setting ICEC=1 in the first &D Namelist, setting up the REACTANT input and inputting values in the FARRAY and OARRAY arrays.

The FARRAY array is used for DUCTs or burners to tell the CEC program the relative amounts of each reactant to be added for fuel. The FARRAY and OARRAY arrays are used in the gas generator component to tell the relative amount of each reactant to add as fuel and oxidizer, respectively. These inputs will be explained further in the following section.



Order	Contents	Format	Column
First line	REACTANTS	3A4	1-9
Next lines	One input line for each reactant species (maximum 15). Air must be the first reactant. Each input line contains:		
	(1) Atomic symbols and formula numbers (maximum 5 sets) <sup>a</sup>	5(A2,F7.5)	1-45
	(2) Blank		46-53
	(3) Enthalpy or heat of combustion <sup>b</sup>	F9.5	54-62
	(4) State: S, L, or G for solid, liquid, or gas, respectively	A1	63
	(5) Temperature, in Kelvin, associated with enthalpy in (3)	F8.5	64-70
	(6) Blank, J, or B <sup>b</sup>	A1	71
	(7) Blank		72
	(8) Combustion efficiency (default is 1.0)	F8.5	73-80
Last line	Blank		
	a. Program will calculate enthalpy or internal energy (3) for species in the thermodynamic data set at the temperature (5) if zeros are in columns 37-38.		
	b. Enthalpy if column 71 is blank or J. Units are cal/g-mole if blank or J/formula wt. if J. Heat of combustion is in BTU/lb if column 71 is B.		

Table 8. Reactant Input

#### 10.4.1 Using The CEC Debugging Parameters

The CEC program has parameters that can be used to debug the operation of the NNEP89 subroutines, the CEC subroutines, or both. These parameters are IDBUG, NDBUG, LDEBUG, and IDEBUG. To use these parameters, they must be set to a non-zero value and input in the CEC &INPT2 Namelist. The values and information printed out to the output dataset for IDBUG are listed in Table 9. The values and output for the other parameters are listed in reference 4. IDBUG is the major debug parameter for the NNEP89 code. NDBUG and LDEBUG are for debugging the THERME subroutine, the interface between the NNEP89 code and the CEC subroutines. IDEBUG is a CEC debug parameter and is specifically for debugging the CEC subroutines. Since the CEC subroutines are called many times during NNEP89 execution, if IDEBUG=1, a lot of IDEBUG information will be produced (At least 100 times more than a "normal" output).

IDBUG	=0	No debug output
	=1	Subroutine names, component and flow station numbers
	=2	Output for IDBUG=1 Mass flows for CEC subroutines calls
	=3	Output for IDBUG=2 Convergence information for burners and gas generators

Table 9. Debug Output for Variable IDBUG

#### 10.4.2 Example Case Using CEC

The sample engine is shown in Figure 8 and is the same as Sample 4 in Section 12.4. The CEC input for Sample 4 is described here in detail to help explain the use of the CEC option. The entire input file is given in Section 12.4.

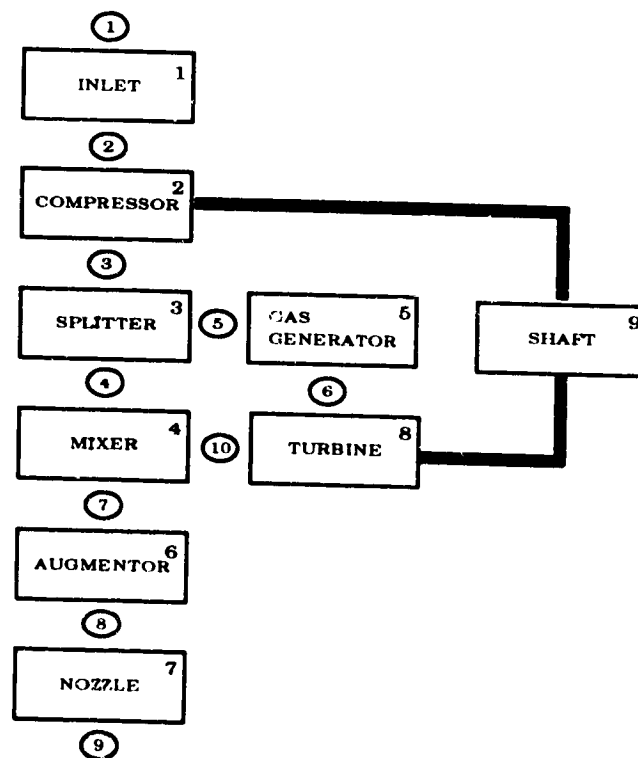


Figure 8. Block Diagram of Air-Turbo Ramjet

The inlet uses "standard air" (assumed to be 75.46% nitrogen, 23.19% oxygen, and 1.35% argon by weight or  $N_{1.5606}O_{0.4198}Ar_{0.0098}$  by atomic formula). The gas generator uses a mixture of liquid and gaseous methane ( $CH_4$ ) for the fuel and liquid and gaseous oxygen ( $O_2$ ) for the oxidizer. The second burner (augmenter) operates at stoichiometric conditions using a mixture of unsymmetrical dimethylhydrazine (atomic formula  $C_2H_8N_2$ ), a typical jet fuel (with a hydrogen to carbon ratio of 0.163, atomic formula  $CH_{1.9423}$ , roughly JP4) and a second typical jet fuel (with a hydrogen to carbon ratio of 0.161, atomic formula  $CH_{1.9185}$ , roughly JP5). The mass ratio of liquid to gaseous methane in the gas generator fuel stream is 6 to 1. The gas generator oxidizer stream is 82% liquid oxygen and 18% gaseous oxygen. The mass fraction of unsymmetrical dimethylhydrazine in the second burner is 0.40, for JP4 it is 0.30, and for JP5 it is the remaining 0.30. All ratios, fractions or percentages in the FARRAY or OARRAY array are by mass. The assigned enthalpy for liquid methane is -21390 cal/mole; for liquid oxygen it is -3102 cal/mole; and for unsymmetrical dimethylhydrazine it is 11900 cal/mole, using the same enthalpy base as discussed in reference 16. The reactant input will be configured to make the CEC routines calculate the correct enthalpy for gaseous oxygen and methane at room temperature. The program can calculate the enthalpies for the input reactants, but only if the exact specie, in the same state (gas, liquid or solid), is in the thermodynamic data. The values for quite a few typical oxidants and fuels can be found in Table VII of reference 16. The heating value for JP4 is 18300 BTU/lb and 18600 BTU/lb for JP5, with burner efficiencies of 0.98 for both compounds; the reactant input for these species will be configured to convert the input heating value and burner efficiency to an equivalent enthalpy.

The reactant input tells the program the relative number of atoms of each element in each specie. After the first &D Namelist input, the next lines will be information for the CEC program. If thermodynamic data are to be included in the input set, that information would be put starting on the first line after the first &D input. The next line would be the word REACTANT (starting in column 1). The following lines would be the reactant information for each species. Because of the setup of the CEC routines, air must be the first reactant, whether the user wants the cycle to use air or not. This example has 8 reactant species. The combustion efficiency and heating value from the input for any burners is ignored when the chemical equilibrium option is used, because the CEC routines set or calculate each specie's enthalpy when it is read in. To have the program calculate the enthalpy of a reactant specie, the user must input 00 in columns 37-38, put the specie's state in column 63, and the specie's temperature, in Kelvin, in columns 64-70. The program will calculate the specie's enthalpy if that exact specie exists in the thermodynamic data; otherwise the program will print an error message, and set the enthalpy to zero. The reactant input for the example case is shown in Figure 9 below. There are column numbers listed above the input. They are there to show the placement of the important terms, they would not be included in the input dataset. The order of the species is somewhat arbitrary, except that the first one must be air. The species are listed in the order that they are mentioned above. There must be a blank line after the last reactant specie input line, and then a CEC switch such as Namelist if there is to be Namelist input following the reactant input or END if that is the end of the CEC input. This example contains no more CEC input, so the reactant input is followed by a blank line and an "END".

	1	2	3	4	5	6	7	8
1234567890123456789012345678901234567890123456789012345678901234567890								
REACTANTS								
N 1.5606 O 0.4198 AR .0098								
C 1.0000 H 4.					100.0	-28.2	G 298.15	O
C 1. H 4.					100.0	-21390.	L 111.66	F
O 2.				00	100.0		G 298.15	F
O 2.				00	100.0	-3102.	L 90.18	O
C 2. H 8. N 2.					100.0		G 298.15	O
C 1. H 1.9423					100.0	11900.	L 298.15	F
C 1. H 1.9185					100.0	18300.	L 296.15BF	0.98
					100.0	18600.	L 298.15BF	0.98
END								

Figure 9. Example REACTANT Input Information

The FARRAY and OARRAY arrays are used to tell the program the relative amounts of each specie used and where it will be used. The FARRAY array is used to tell the relative amount of each specie to be added as the fuel. The FARRAY array is only required for burners and gas generators. The OARRAY array is used to tell the relative amount of each specie to be added as the oxidizer, and is only required for gas generators. The FARRAY and OARRAY arrays are very flexible, a specie that is in a OARRAY for one component can be in the FARRAY for another. The same reactant specie could even be in both the FARRAY and OARRAY array in the gas generator component, if necessary. FARRAY and OARRAY are 2-dimensional arrays, (i,j). The j index corresponds with the component number of the burner or gas generator. The i index contains pairs of reactant number and relative mass. It is input as a reactant number, its relative amount, second reactant number, its relative number, and so on. Up to 6 different reactants can be included in each FARRAY or OARRAY array. A reactant number is given to each reactant specie as it is read in from the input. For the given example, air is 1, liquid methane is 2, gaseous methane is 3, and so on. Whole numbers or fractions can be used for the relative mass indices. The program will sum and normalize the relative mass numbers internally. For the example, the FARRAY and OARRAY's become:

```

FARRAY (1,5)=2, 6, 3, 1,
OARRAY (1,5)=4, 82, 5, 18,
FARRAY (1,6)=6, .4, 7, .3, 8, .3,

```

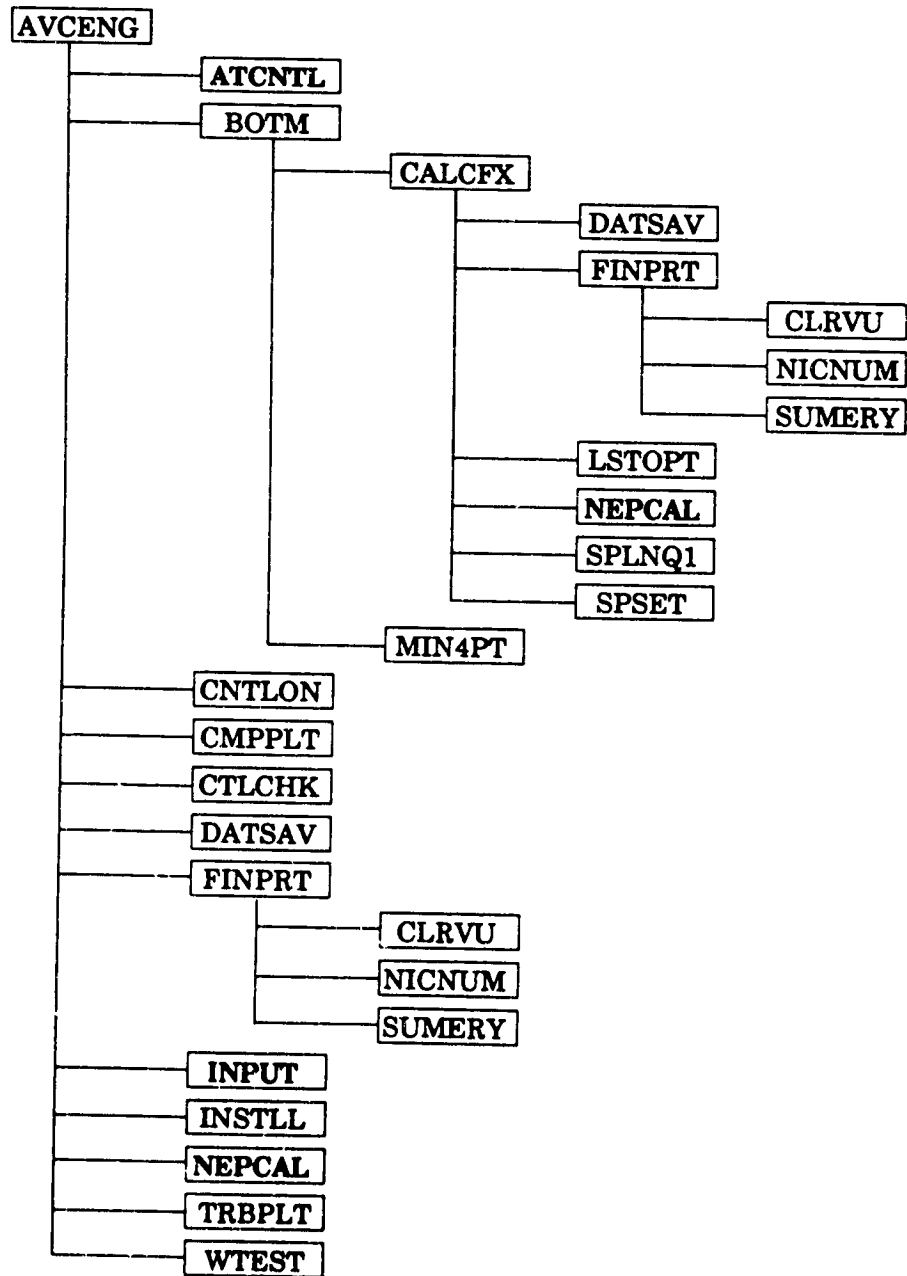
The FARRAY and OARRAY arrays are included in the &D Namelist input and are generally input with the rest of the component inputs. This is shown in Sample 4 in Section 12.4.

#### 10.4.3 Using The IDBUG Option Without Using CEC Option

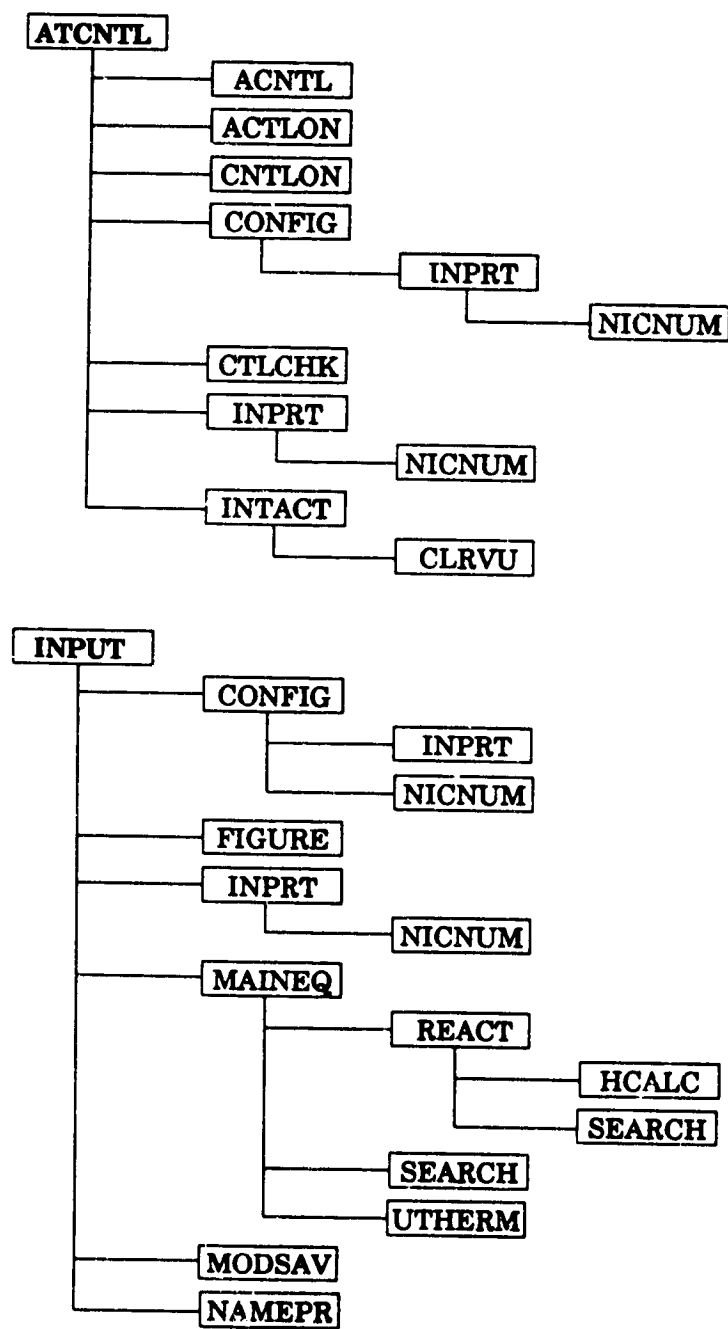
It is possible to use the debug parameter IDBUG (discussed previously and in reference 4) without using the chemical equilibrium option. This is done by setting ICEC=1 in the global &D Namelist input. In the CEC &INPT2 Namelist input, set variables ICEC=0 and IDBUG to the desired value from Table 9. This option has been found to be very useful for determining errors in input datasets by printing intermediate output during execution. The first few lines of the input dataset would look like the following:

```
TITLE CARD
&D ICEC=1 (and other global &D Namelist variables)
&END
NAMELIST
&INPT2 IDBUG=1, ICEC=0, &END
END
&D (KONFIG and SPEC input)
.
.
&END
```

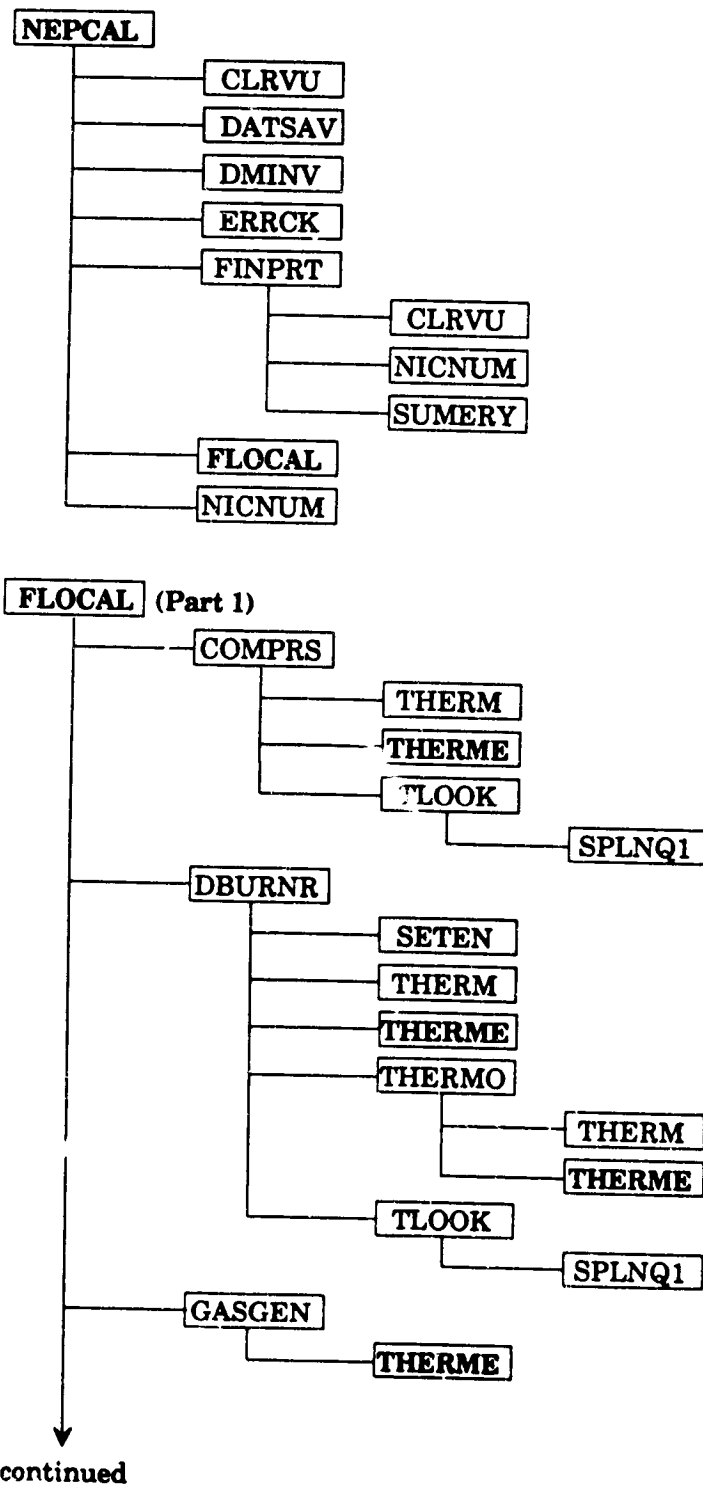
**11.0 NNEP89 FLOW-CHART**  
**(BOLD indicates new branch)**



NNEP89 Flow-Chart (2)

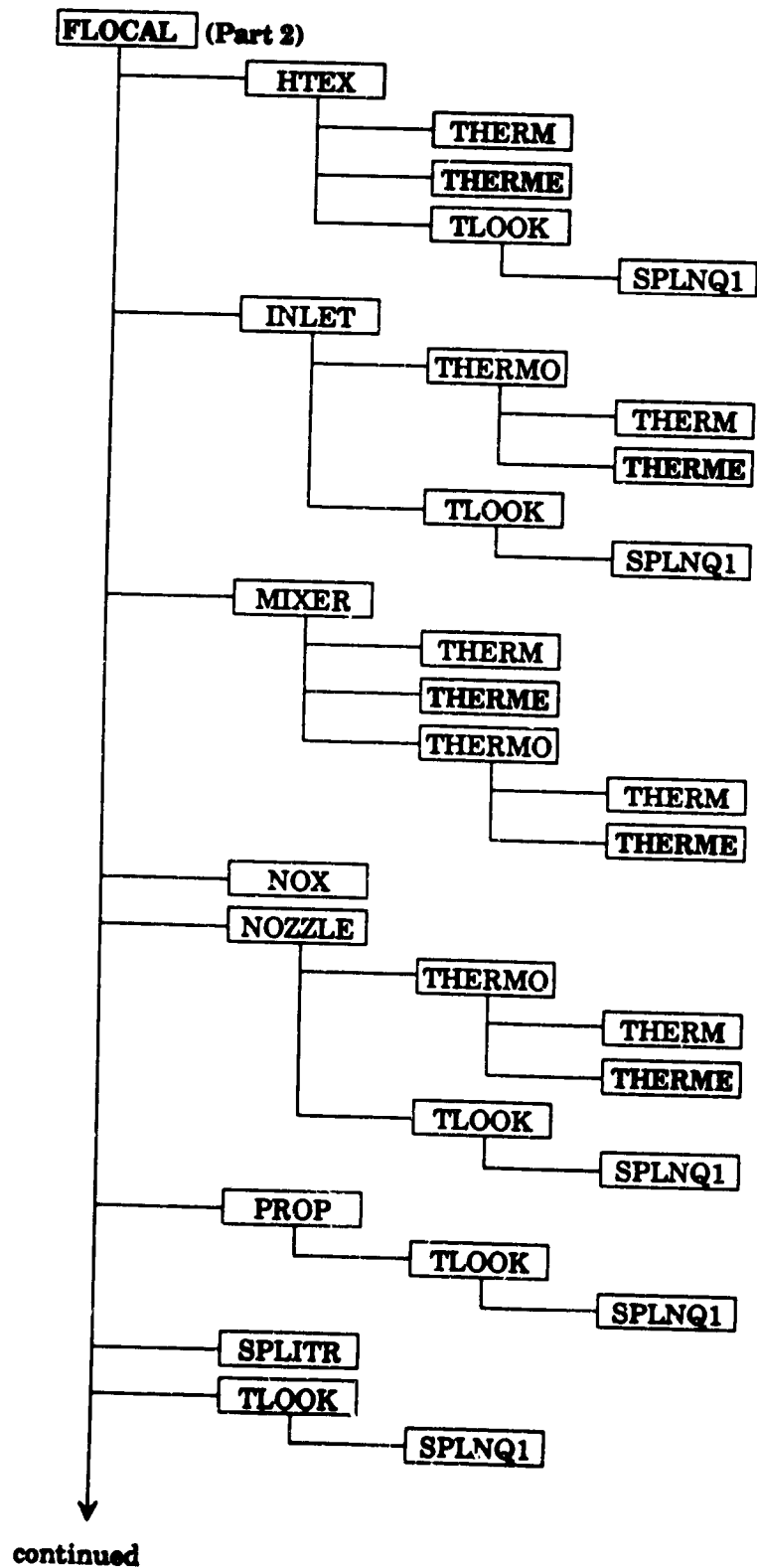


# NNEP89 Flow-Chart (3)

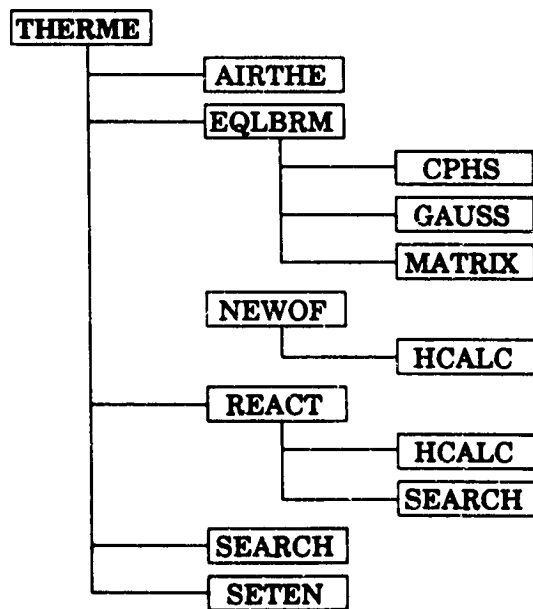
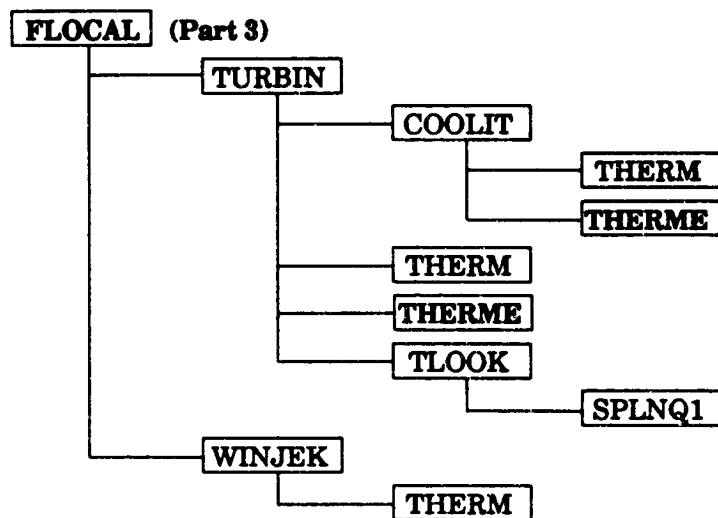




NNEP89 Flow-Chart (4)



NNEP89 Flow-Chart (5)



## 12.0 SAMPLE INPUT AND OUTPUT

The following sections list the input files for five different engine configurations. The configurations and options used in these samples were chosen to demonstrate the wide range of capabilities of the NNEP89 code. A brief description of the engine configuration being modeled and the options being used precedes each input file. A block diagram is also included to aid the reader in understanding the configuration being modeled. The program output that corresponds to the third sample input is also given. Finally, the compressor and turbine map tables that were used in all the examples are given.

## 12.1 SAMPLE 1 - TWO-SPOOL MIXED-FLOW TURBOFAN.

Sample 1 is a 2-spool, mixed-flow turbofan using the default thermodynamics routines. NNEP89 will draw a block diagram of the engine components (DRAW=T), and print information (number of iterations, altitude, Mach number, inlet recovery, engine airflow, gross thrust, fuel flow, net thrust, and TSFC) to the terminal as the case is executing (ITERM=2). A block diagram of this engine is shown in Figure 10. This case also includes input that is commented out using /\* and \*/.

```

SAMPLE MIXED FLOW TURBOFAN
&D ITERM=2, DRAW=T, LONG=T, TABLES=T, PINPUT=T &END
&D MODE=1
KONFIG(1,1)=4HINLT,1,0,2,0,SPEC(1,1)=100.0,0.,0.,0.,0.,
0.,0.,0.,0.,0.,0.,18.00,
KONFIG(1,2)=4HCOMP,2,0,3,0,SPEC(1,2)=1.500,0.,1.0,1001.,1.0,
1002.,1.0,1003.,1.0,0.,0.,0.8500,3.0,1.0,
KONFIG(1,3)=4HSPLT,3,0,4,9,SPEC(1,3)=1.0,0.2000E-01,0.2000E-01,
KONFIG(1,4)=4HCOMP,4,0,5,12,SPEC(1,4)=1.300,0.5000E-01,1.0,1004.,1.0,
1005.,1.0,1006.,1.0,0.,0.,0.8600,6.0,1.0,
KONFIG(1,5)=4HDUCT,5,0,6,0,SPEC(1,5)=0.5000E-01,0.3000,0.,3000.,0.9900,
18300.,0.,0.,0.,0.5000E-01,
KONFIG(1,6)=4HTURB,6,12,7,0,SPEC(1,6)=3.500,0.7500,1.0,1007.,1.0,
1008.,1.0,1.0,0.8000,1.0,0.9000,5000.,1.0,
KONFIG(1,7)=4HTURB,7,12,8,0,SPEC(1,7)=2.500,0.2500,1.0,1009.,1.0,
1010.,1.0,1.0,1.0,0.9000,5000.,1.0,
KONFIG(1,8)=4HMIXR,8,9,10,0,SPEC(1,8)=0.,0.,0.4000,1.0,
KONFIG(1,9)=4HNOZZ,10,0,11,0,SPEC(1,9)=0.,0.9800,0.,0.,0.9750,
1.0,0.,0.,1.0,
KONFIG(1,10)=4HLOAD,0,0,0,0,SPEC(1,10)=-200.0,
KONFIG(1,11)=4HSHFT,4,6,10,0,SPEC(1,11)=8000.,8*1,
KONFIG(1,12)=4HSHFT,2,7,0,0,SPEC(1,12)=6000.,8*1,1.0,0.,0.,
KONFIG(1,13)=4HCNTL,SPCNTL(1,13)=1.0,7.0,4HSTAP,8.0,10,0,1,
KONFIG(1,14)=4HCNTL,SPCNTL(1,14)=1.0,6.0,4HSTAP,8.0,7,0,1,
KONFIG(1,15)=4HCNTL,SPCNTL(1,15)=1.0,4.0,4HSTAP,8.0,6,0,1,1.1,1.75,
KONFIG(1,16)=4HCNTL,SPCNTL(1,16)=1.0,3.0,4HSTAP,8.0,4,0,1,
KONFIG(1,17)=4HCNTL,SPCNTL(1,17)=1.0,1.0,4HSTAP,8.0,2,0,1,
KONFIG(1,18)=4HCNTL,SPCNTL(1,18)=1.0,2.0,4HDOUT,8.0,8,0,1,1.1,2.1,
KONFIG(1,19)=4HCNTL,SPCNTL(1,19)=1.0,11.00,4HDOUT,8.0,11,0,1,
KONFIG(1,20)=4HCNTL,SPCNTL(1,20)=1.0,12.00,4HDOUT,8.0,12,0,1,
&END
&D MACH= 0.30,ALTP= 5000.,ETAR=1.0000,MODE= 1, &END
/* &D MACH= 0.30,ALTP= 10000.,ETAR=1.0000,MODE= 1, &END
&D MACH= 0.30,ALTP= 15000.,ETAR=1.0000,MODE= 1, &END
&D MACH= 0.30,ALTP= 20000.,ETAR=1.0000,MODE= 1, &END
&D MACH= 0.30,ALTP= 25000.,ETAR=1.0000,MODE= 1, &END */
&D MACH= 0.50,ALTP= 15000.,ETAR=1.0000,MODE= 1, &END
&D MACH= 0.70,ALTP= 25000.,ETAR=1.0000,MODE= 1, &END
&D MACH= 0.80,ALTP= 40000.,ETAR=1.0000,MODE= 1, &END
&D MACH= 1.20,ALTP= 40000.,ETAR=0.9915,MODE= 1, &END
&D MACH= 1.40,ALTP= 40000.,ETAR=0.9782,MODE= 1, &END
&D MACH= 1.80,ALTP= 40000.,ETAR=0.9445,MODE= 1, &END

```

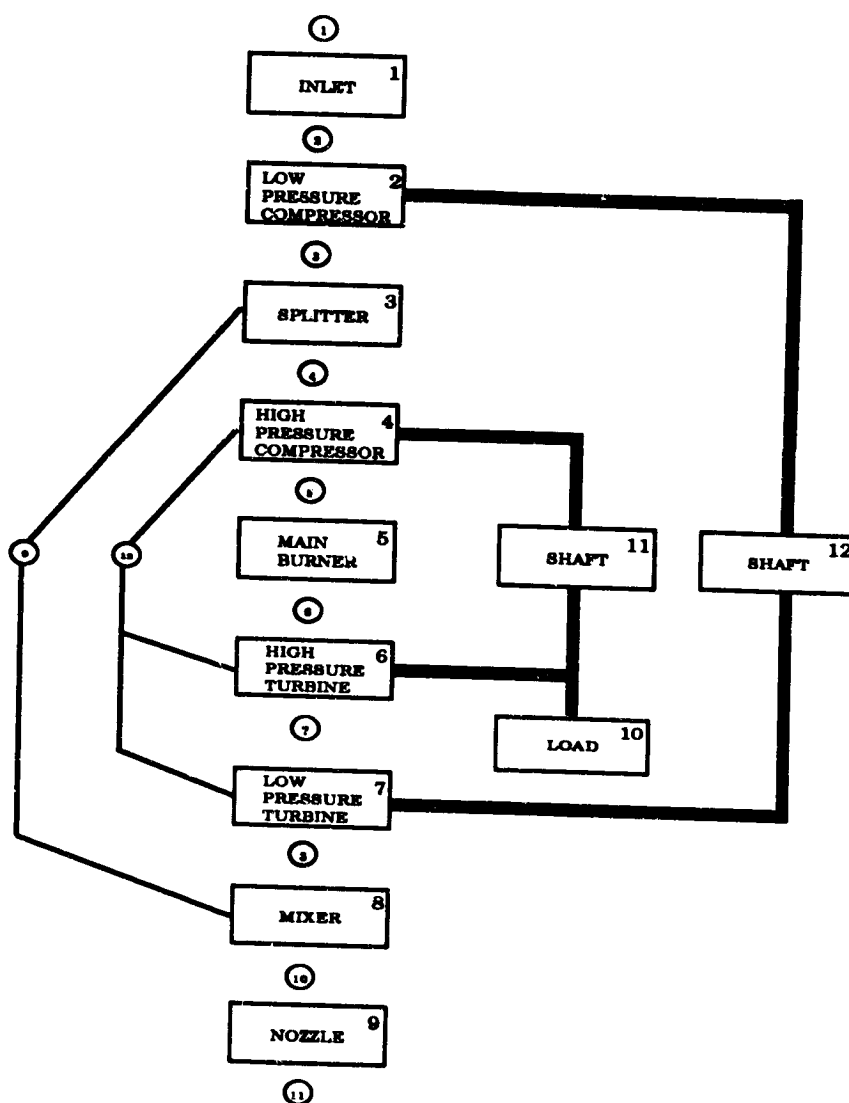


Figure 10. Block Diagram of Two-Spool, Mixed-Flow Turbofan

## 12.2 SAMPLE 2 - SEPARATE FLOW TURBOFAN USING CEC.

Sample 2 is a 1-spool, separate-flow turbofan (with regenerator) using the CEC thermodynamics routines (ICEC=1) and automatic controls (ACTL=1) to take care of all flow and work errors. The bypass air is being heated by the core flow before the core flow exits through a nozzle. The fuel is a "standard" JP fuel with a heating value of 18300.BTU/lb and a burner efficiency of 0.99. A block diagram of this engine in shown in Figure 11.

```

TEST CASE: HTEX SANFORD NEW
&D ACTL=1, ICEC=1, LONG=F, NCODE=1, DRAW=T, NMODES=1, &END
REACTANTS
N 1.5606 O 0.4198 AR .0098          100.    0.0    G 298.15 O
C 0.5245 H 1.0000                  100.   -4628.066L 298.15 F

NAMELIST
&INPT2 ICEC=1,&END
END
&D MODE=1,
KONFIG(1,1)=4HINLT,1,0,2,0,SPEC(1,1)=5*0,.97,5*0,14.4,0,100,
KONFIG(1,2)=4HCOMP,2,0,3,12,SPEC(1,2)=1.3,.001,1,1004,1,1005,1,1006,
1,0,0,.85,12,0,1,
KONFIG(1,3)=4HSPLT,3,0,9,4,SPEC(1,3)=20.,0,0,
KONFIG(1,4)=4HDUCT,4,0,5,0,SPEC(1,4)=.05,0,0,3200.,.99,18300,
FARRAY(1,4)=2.,1.,
KONFIG(1,5)=4HTURB,5,12,6,0,SPEC(1,5)=3.3,1,1,1007,1,1008,1,1,1.,
1.,.91,5000,1,
KONFIG(1,6)=4HHTEX,9,6,10,7,SPEC(1,6)=.05,.03,800,.8,1,
KONFIG(1,7)=4HNOZZ,7,0,8,0,SPEC(1,7)=0,1,0,0,0.9850,1,1,0,1,
KONFIG(1,8)=4HSHFT,2,5,0,0,SPEC(1,8)=9*1.C
KONFIG(1,9)=4HNOZZ,10,0,11,0,SPEC(1,9)=0,1,0,0,0.9850,1,1,0,1,
&END
&D MACH= 0.30,ALTP= 5000.,ETAR=1.0000,MODE= 1, &END
&D MACH= 0.50,ALTP= 15000.,ETAR=1.0000,MODE= 1, &END

```

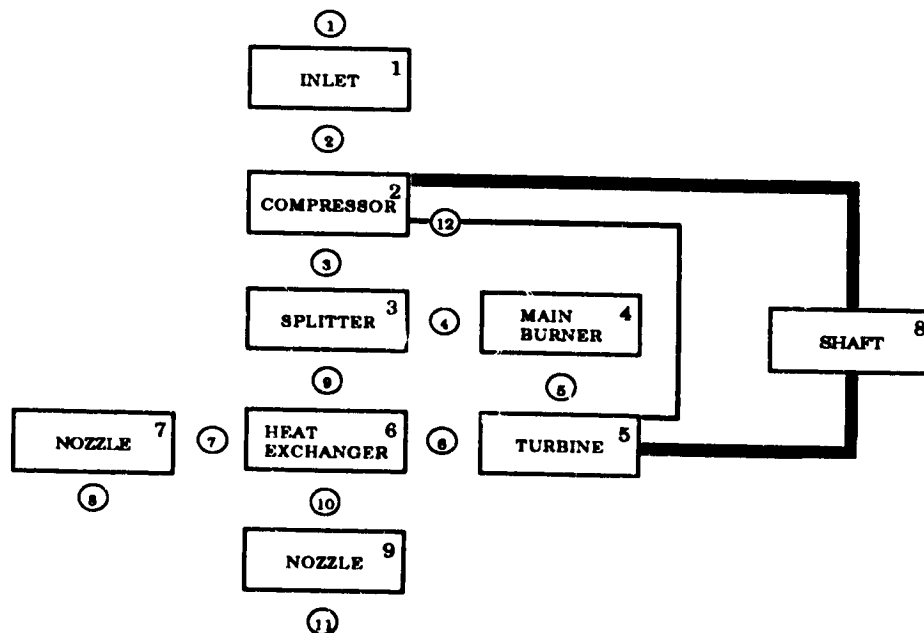


Figure 11. Block Diagram of One-Spool, Separate-Flow Turbofan (with Regenerator)

### 12.3 SAMPLE 3 - ONE-SPOOL MIXED-FLOW TURBOFAN

Sample 3. is a 1-spool, mixed-flow turbfan using the default thermodynamics routines and output of the convergence history (LONG=T). This case also uses the XNUM array to name the flow stations on the output, but XNUM does not work on systems that cannot handle Namelist Hollerith or quoted string input. Removing the XNUM array would not cause any problems executing this input case. This case also has the headings option on (DOUTHDT=T). (Prints out headers on the output file to aid the user interpreting the output, highly recommended for novice users.) The program will draw a block diagram of the engine (DRAW=T) and will make full passes through the engine while it determines the error matrix (NCODE=-1). A block diagram of this engine is shown in Figure 12.

```
TEST CASE: MIXED FLOW TURBOFAN SANFORD NEW
&D LONG=T, NCODE=-1, DOUTHDT=T, DRAW=T, &END
&D MODE=1,
XNUM=4HINLT, 4HLPCI, 4HSPLI, 4HHPCI, 4HMBRN, 4HHPTI, 4HMXRI, 4HABRN,
4HNOZI, 4HNOZO, 4HSECI, 4HSECO,
KONFIG(1,1)=4HINLT, 1, 0, 2, 0, SPEC(1,1)=100.0, 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 18.00,
KONFIG(1,2)=4HCOMP, 2, 0, 3, 0, SPEC(1,2)=1.60, 0., 1, 1001, 1, 1002, 1,
1003, 1, 0., 0., 0.84, 2.9, 0.95,
KONFIG(1,3)=4HSPLT, 3, 0, 4, 12, SPEC(1,3)=1.75, 0.02, 0.02,
KONFIG(1,4)=4HCOMP, 4, 0, 5, 0, SPEC(1,4)=1.30, 0., 1., 1004, 1, 1005, 1,
1006, 1, 0., 0., 0.84, 6.0, 1.,
KONFIG(1,5)=4HDUCT, 5, 0, 6, 0, SPEC(1,5)=-.05, 0, 0, 3000., .99, 18300.,
FARRAY(1,5)=2, 1,
KONFIG(1,6)=4HTURB, 6, 0, 7, 0, SPEC(1,6)=3.6, 1.0, 1, 1007, 1, 1008,
1, 1, 1, 1., .90, 5000, 1,
KONFIG(1,7)=4HMIXR, 7, 13, 8, 0, SPEC(1,7)=0., 0., 0.40, 1., 0., 0.,
KONFIG(1,8)=4HDUCT, 8, 0, 9, 0, SPEC(1,8)=-.05, 0, 0, 3600., .99, 18300.,
FARRAY(1,8)=2, 1,
KONFIG(1,9)=4HNOZZ, 9, 0, 10, 0, SPEC(1,9)=0., 0.98, 0., 0., .98, 1., 1.,
0., 1.,
KONFIG(1,10)=4HDUCT, 12, 0, 13, 0, SPEC(1,10)=-.05, 0, 0, 0., 0.99, 18300.,
FARRAY(1,10)=2, 1,
KONFIG(1,11)=4HSHFT, 2, 4, 6, 0, SPEC(1,11)=9*1.0,
KONFIG(1,12)=4HCNTL, SPCNTL(1,12)=1, 1, 4HSTAP, 8, 2, 0, 0,
KONFIG(1,13)=4HCNTL, SPCNTL(1,13)=1, 4, 4HSTAP, 8, 4, 0, 0,
KONFIG(1,14)=4HCNTL, SPCNTL(1,14)=1, 11, 4HDOUT, 8, 11, 0, 0,
KONFIG(1,15)=4HCNTL, SPCNTL(1,15)=1, 6, 4HSTAP, 8, 6, 0, 0,
KONFIG(1,16)=4HCNTL, SPCNTL(1,16)=1, 3, 4HDOUT, 8, 7, 0, 0,
KONFIG(1,17)=4HCNTL, SPCNTL(1,17)=1, 2, 4HDOUT, 5, 2, 50, 1,
&END
&D SPEC(9,12)=1, SPEC(9,13)=1, SPEC(9,14)=1, SPEC(9,15)=1,
SPEC(5,1)=-.26, SPEC(9,1)=50, SPEC(6,1)=0,
SPEC(9,16)=1, SPEC(9,17)=1, &END
```

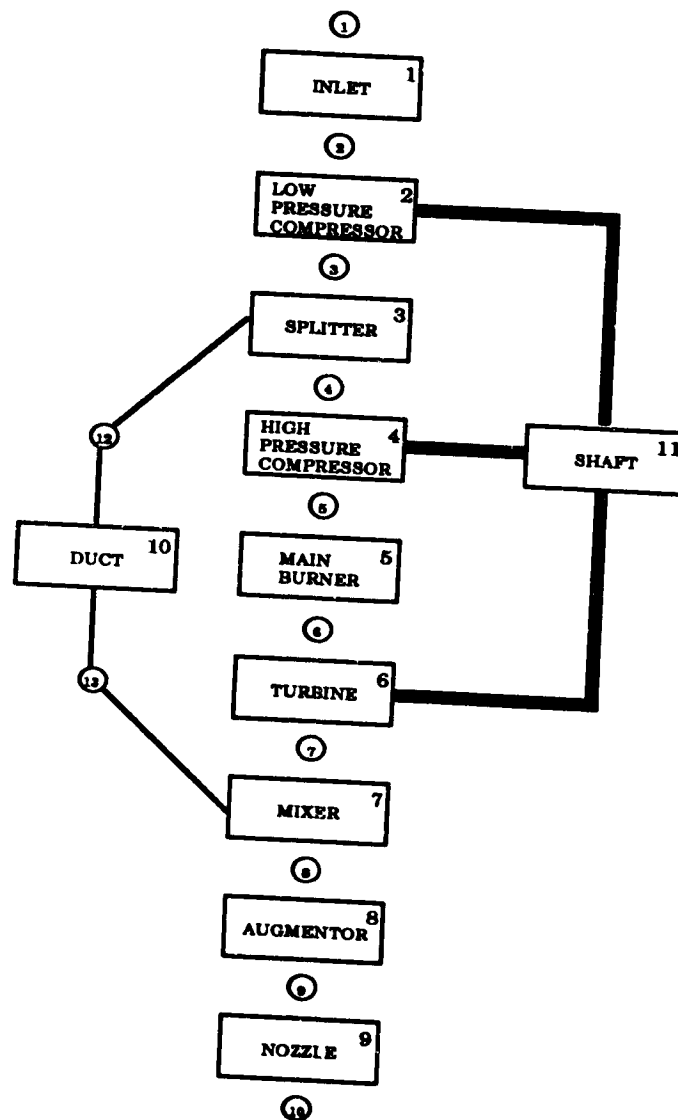


Figure 12. Block Diagram of One-Spool, Mixed-Flow Turbofan



## 12.4 SAMPLE 4 - AIR-TURBO RAMJET USING CEC

Sample 4. is a 1-spool, gas-generator Air-Turbo Ramjet, using the CEC thermodynamic routines (ICEC=1). NNEP89 will draw a block diagram of the cycle (DRAW=T) and print a little information about the convergence history (LONG=T). The code will make a full pass through the cycle when it calculates the partial derivative matrix (NCODE=-1). The gas-generator operates on liquid and gaseous methane for the fuel and liquid and gaseous oxygen for oxidizer, combustion occurs at equivalence ratios greater than stoichiometric. The cycle has a stoichiometric augmentor fuelled with a mixture of unsymmetrical dimethylhydrazine ( $C_2H_8N_2$ ) and 2 hydrocarbon species similar to JP4 and JP5. The CEC subroutines will determine the enthalpies for the hydrocarbon species from the lower heating value input in that reactant line. The enthalpy for gaseous methane and oxygen will be calculated by the program because of the 00 in columns 37-38. This input file also contains Namelist input for the CEC program, with the variable IDBUG. Since IDBUG=0, debugging is turned off, but setting IDBUG to 1, 2, or 3 would print our additional information that could be very useful for finding errors when an input case will not execute. Caution is suggested when using the CEC debug parameters, it increases program output by a factor of about 10 or more. A block diagram of this engine in shown i Figure 13.

```

ATR TO DEMONSTRATE THE USE OF CEC, GAS GENERATORS AND FARRAYS
&D ICEC=1, DRAW=T, LONG=T, NCODE=-1, &END
REACTANTS
N 1.5606 O 0.4198 AR .0098
C 1.      H 4.
C 1.      H 4.      00
O 2.
O 2.      00
C 2.      H 8.      N 2.0
C 1.      H 1.9423
C 1.      H 1.9185
1.      -28.2      G 298.15 O
100.0 -21390.      L 111.66 F
100.0      G 298.15 F
100.      -3102.      L 90.18 O
100.      G 298.15 O
100.0 11900.      L 298.15 F
100.0 18300.      L 298.15BF 0.98
100.0 18600.      L 298.15BF 0.98

NAMELIST
&INPT2 IDBUG=0, &END
END
&D MODE=1,
KONFIG(1,1)=1,1,0,2,0,SPEC(1,1)=100,4*0,.97,2*0,.1,2*0,14,
KONFIG(1,2)=4,2,0,3,0,SPEC(1,2)=1.6,0,1,1001,1,1002,1,1003,
1,0,0,.85,2,0,1,
KONFIG(1,3)=7,3,0,4,5,SPEC(1,3)=0.,0.,
KONFIG(1,4)=8,4,10,7,0,SPEC(1,4)=0.,0.,0.25,.95,
KONFIG(1,5)=3,5,0,6,0,SPEC(1,5)=2960,.67184,600.,1.644,
FARRAY(1,5)=2,6,3,1,
OARRAY(1,5)=4,82,5,18,
KONFIG(1,6)=2,7,0,8,0,SPEC(1,6)=.00001,0,0,0,0,95,21500.,
SPEC(1,6)=1,
FARRAY(1,6)=6.,4,7.,3,8.,3,
KONFIG(1,7)=9,8,0,9,0,SPEC(1,7)=0,1,0,0,.985,1,1,0,1,
KONFIG(1,8)=5,6,0,10,0,SPEC(1,8)=3.6,0.,1,1007.,1,
1008,1,1,0,1,.80,5000,1,
KONFIG(1,9)=11,2,8,0,0,SPEC(1,9)=9*1,
KONFIG(1,10)=12,SPCNTL(1,10)=1.0,1.0,100,8.0,2.0,0.,0,0,0.,
KONFIG(1,11)=12,SPCNTL(1,11)=4.0,5.0,200,8.0,4.0,0,0,0,0.,
KONFIG(1,12)=12,SPCNTL(1,12)=3.0,5.0,100,8.0,6.0,0.,0,0,0.,
KONFIG(1,13)=12,SPCNTL(1,13)=1.0,8.0,200,8.0,9.0,0.,0,0,0.,
KONFIG(1,14)=12,SPCNTL(1,14)=1.0,9.0,200,5.0,2.0,20.,0,0,0.,
KONFIG(1,15)=12,SPCNTL(1,15)=4.0,5.0,100,2.0,10.0,28.510,0,0,0,
&END
&D SPEC(9,10)=1,SPEC(9,11)=1,SPEC(9,12)=1,SPEC(9,13)=1,
SPEC(5,1)=.26,SPEC(9,1)=50,SPEC(6,1)=0, &END
&D SPEC(5,1)=.6,SPEC(9,1)=10000, &END

```

```

&D SPEC(5,1)=-.9,SPEC(9,1)=25000, &END
&D SPEC(5,1)=-1.2,SPEC(9,1)=32200, &END
&D SPEC(5,1)=-1.5,SPEC(9,1)=39500, &END
&D SPEC(5,1)=-1.8,SPEC(9,1)=46800, &END
&D SPEC(5,1)=-2.0,SPEC(9,1)=50900, &END
&D SPEC(5,1)=-2.5,SPEC(9,1)=55000, &END
&D SPEC(5,1)=-3.0,SPEC(9,1)=58400, &END

```

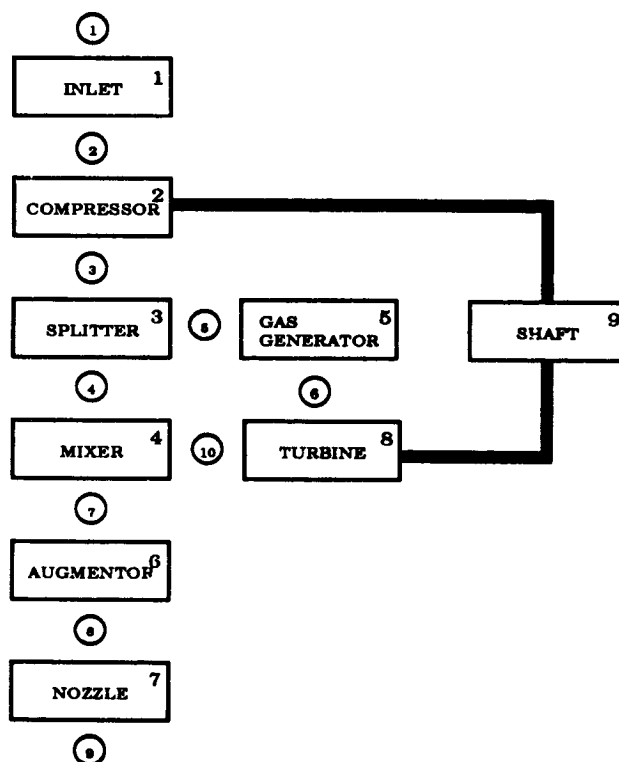


Figure 13. Block Diagram of Air-Turbo Ramjet

## 12.5 SAMPLE 5 - TANDEM FAN WITH MULTIMODES.

Sample 5. is a 2-spool, tandem fan engine, with multimode capabilities. Mode 1 is in series, mixed flow, mode 2 in parallel-flow, high bypass. The input is set to draw a block diagram of each mode (DRAW=T), and the mode to design the components is mode 1 (MODESN=1). This is a 2 mode engine (NMODES=2) and will make full passes through the engine while it determines the error matrix (NCODE=-1). The data output headers option is on (DOUTHD=T). (Prints out headers on the output file to aid the user interpreting the output, highly recommended for novice users.) This case will print information (number of iterations, altitude, Mach number, inlet recovery, engine airflow, gross thrust, fuel flow, net thrust, and TSFC) to the terminal as the case is executing (ITERM=2). This engine uses XNUM to label the station property outputs, the XNUM array could be removed without changing the way the engine is operated. Since this engine may require more than 50 iterations to converge, the maximum number of iterations has been increased to 75 (MAXNIT=75). A block diagrams of this engine in modes 1 and 2 are shown in Figures 14 and 15, respectively.

```
2 MODE TANDEM-FAN ENGINE 1=SERIES,MXDFLO 2=PARALLEL HY-BY
&D NCODE=-1,DRAW=T, LONG=F,NMODES=2,MODESN=1,DOUTHD=T, &END
&D MODE=1,ITERM=2. MAXNIT=75,
XNUM=4HINTI,4HDSPI,4HFFNI,4HFFNO,4H2FNI,4H2FNO,4HHPCI,4HHPCO,4HHPTI,
4HLPTI,4HMXMI,4HMIXO,4HNOZI,4HNOZO,4HDDTI,4HDNZI,4HDNZO,4HCOOL,4HBDUC,
4HMXSI,
KONFIG(1,1)=4HINLT,1,0,2,0,SPEC(1,1)=300.0,0.,0.,0.,0.,1.0,
KONFIG(1,2)=4HCOMP,3,0,4,0,SPEC(1,2)=1.5,0.,1.0,1001.,1.0,
1002.,1.0,1003.,1.0,00.00,0.,0.8800,2.800,1.0,
KONFIG(1,3)=4HCOMP,5,0,6,0,SPEC(1,3)=1.5,0.,1.0,1001.,1.0,
1002.,1.0,1003.,1.0,00.00,0.,0.8800,1.500,0.95,
KONFIG(1,4)=4HSPLT,6,0,7,19,SPEC(1,4)=0.7000,0.2000D-01,0.2000D-01,
KONFIG(1,5)=4HCOMP,7,0,8,18,SPEC(1,5)=1.3,0.1000D00,1.0,1004.,1.0,
1005.,1.0,1006.,1.0,0.,0.,0.8800,7.200,0.935,
KONFIG(1,6)=4HDUCT,8,0,9,0,SPEC(1,6)=0.6000D-01,0.,0.,3260.,0.9850,
0.1830D05,0.,0.,0.,0.1000D00,
KONFIG(1,7)=4HSPLT,2,0,3,15,SPEC(1,7)=0.1000D-01,
KONFIG(1,8)=4HTURB,9,18,10,0,SPEC(1,8)=3.500,0.7000,1.0,1007,1.0,
1008,1.0,1.0,0.5000,1.0,0.9000,5000.,1.0,
KONFIG(1,9)=4HTURB,10,18,11,0,SPEC(1,9)=2.500,0.3000,1.0,1009,1.0,
1010,1.0,1.0,0.5000,1.0,0.8800,5000.,1.0,
KONFIG(1,11)=4HDUCT,19,0,20,0,SPEC(1,11)=0.5000D-01,0.,0.,400.0,0.9700,
0.1830D05,0.,0.,0.,0.5000D-01,
KONFIG(1,12)=4HMIXR,11,20,12,0,SPEC(1,12)=0.,0.,0.3000,0.9500,1.0,1.0,
KONFIG(1,14)=4HDUCT,12,0,13,0,SPEC(1,14)=0.5000D-01,0.,0.,400.0,0.9700,
0.1830D05,0.,0.,0.,0.5000D-01,
KONFIG(1,15)=4HNOZZ,13,0,14,0,SPEC(1,15)=0.,1.0,0.,0.,0.9800,1.0,0.,0.,1.0,
KONFIG(1,16)=4HLOAD,0,0,0,0,SPEC(1,16)=-100.0,
KONFIG(1,17)=4HSHFT,5,8,0,0,SPEC(1,17)=0.1400D05,1.0,1.0,1.0,1.0,
1.0,1.0,1.0,1.0,
KONFIG(1,18)=4HSHFT,2,3,16,9,SPEC(1,18)=8000.,1.0,1.0,1.0,1.0,
1.0,1.0,1.0,1.0,
KONFIG(1,19)=4HDUCT,15,0,16,0,SPEC(1,19)=0.5000D-01,0.,0.,400.0,0.9700,
0.1830D05,
KONFIG(1,21)=4HCNTL,SPCNTL(1,21)=1.0,9.0,4HSTAP,8.0,
13.00,0.,1.0,1.0,0.,
KONFIG(1,22)=4HCNTL,SPCNTL(1,22)=1.0,8.0,4HSTAP,8.0,
10.00,0.,1.0,1.0,0.,
KONFIG(1,23)=4HCNTL,SPCNTL(1,23)=1.0,17.00,4HDOUT,8.0,
17.00,0.,1.0,0.,0.,
KONFIG(1,24)=4HCNTL,SPCNTL(1,24)=1.0,18.00,4HDOUT,8.0,
18.00,0.,1.0,0.,0.,
KONFIG(1,25)=4HCNTL,SPCNTL(1,25)=1.0,2.0,4HDOUT,5.0,
2.0,25.00,0.,1.050,4.0,
KONFIG(1,26)=4HCNTL,SPCNTL(1,26)=1.0,3.0,4HDOUT,5.0,
3.0,25.00,0.,1.050,4.0,
KONFIG(1,27)=4HCNTL,SPCNTL(1,27)=1.0,5.0,4HSTAP,8.0,
```

```

9.0,0.,1.0,1.050,4.0,
KONFIG(1,28)=4HCNTL,SPCNTL(1,28)=1.0,4.0,4HDOUT,8.0,
12.00,0.,1.0,0.,0.,
KONFIG(1,29)=4HCNTL,SPCNTL(1,29)=1.0,1.0,4HSTAP,8.0,
3.0,0.,1.0,0.,0.,
KONFIG(1,31)=4HCNTL,SPCNTL(1,31)=10.00,2.0,4HSTAP,8.0,
5.0,0.,1.0,0.,40.00,
KONFIG(1,32)=4HCNTL,SPCNTL(1,32)=10.00,3.0,4HSTAP,8.0,
7.0,0.,1.0,0.,40.00,
KONFIG(1,37)=4HLIMV,0,0,0,0,SPEC(1,37)=0,0,8800.,4HDOUT,2.,18.,0,0,1,
KONFIG(1,40)=4HLIMV,0,0,0,0,SPEC(1,40)=0,0,1.100,4HDOUT,6.,2.,0,0,1,
KONFIG(1,45)=4HLIMV,0,0,0,0,SPEC(1,45)=0,0,1.100,4HDOUT,6.,3.,0,0,1,
KONFIG(1,48)=4HLIMV,0,0,0,0,SPEC(1,48)=0,1.050,4.0,4HDOUT,4.,5.,0,0,1,
KONFIG(1,49)=4HLIMV,0,0,0,0,SPEC(1,49)=0,0,1.100,4HDOUT,6.,5.,0,0,1,
KONFIG(1,50)=4HLIMV,0,0,0,0,SPEC(1,50)=0,2.0,80.00,4HDOUT,5.,5.,0,0,1,
KONFIG(1,54)=4HOPTV,0,0,15,0,SPEC(1,54)=0.,0.,0.,1.0,0.,
0.,0.,0.,1.0,
KONFIG(1,58)=4HCNTL,SPCNTL(1,58)=4.0,6.0,4HDOUT,6.0,
3.0,1.0,1.0,0.,0.,
KONFIG(1,59)=4HDUCT,4,0,5,0,SPEC(1,59)=0.,
KONFIG(1,60)=4HNOZZ,16,0,17,0,SPEC(1,60)=0.,0.1000D00,0.,0.,0.1000D00,
1.0,1.0,0.,1.0,
&END
&D MODE=2,
XNUM=4HINTI,4HSPLI,4HFFNI,4HVLVI,4H2FNI,4H2FNO,4HHPCI,4HHPCO,4HHPTI,
4HLPIT,4HLPOT,4H      ,4HNOZI,4HNOZO,4HFFNO,4HDNZI,4HDNZO,4HCOOL,4HBDUC,
4HBNZI,4HBNZO,
KONFIG(1,1)=4HINLT,1,0,2,0,
KONFIG(1,2)=4HCOMP,3,0,15,0,
KONFIG(1,3)=4HCOMP,5,0,6,0,
KONFIG(1,4)=4HSPLT,6,0,7,19,
KONFIG(1,5)=4HCOMP,7,0,8,18,
KONFIG(1,6)=4HDUCT,8,0,9,0,
KONFIG(1,7)=4HSPLT,2,0,4,3,
KONFIG(1,8)=4HTURB,9,18,10,0,
KONFIG(1,9)=4HTURB,10,18,11,0,
KONFIG(1,11)=4HDUCT,19,0,20,0,
KONFIG(1,13)=4HNOZZ,20,0,21,0,SPEC(1,13)=0.,1.0,0.,0.,0.9500,
1.0,0.,0.,1.0,
KONFIG(1,14)=4HDUCT,11,0,13,0,
KONFIG(1,15)=4HNOZZ,13,0,14,0,
KONFIG(1,16)=4HLOAD,0,0,0,0,
KONFIG(1,17)=4HSHFT,5,8,0,0,
KONFIG(1,18)=4HSHFT,2,3,16,9,
KONFIG(1,19)=4HDUCT,14,0,16,0,
KONFIG(1,20)=4HNOZZ,16,0,17,0,SPEC(1,20)=0.,1.0,0.,0.,0.9800,
1.0,0.,0.,1.0,
KONFIG(1,22)=4HCNTL,SPCNTL(1,22)=1.0,8.0,4HSTAP,8.0,
10.00,0.,1.0,1.0,0.,
KONFIG(1,23)=4HCNTL,SPCNTL(1,23)=1.0,17.00,4HDOUT,8.0,
17.00,0.,1.0,0.,0.,
KONFIG(1,25)=4HCNTL,SPCNTL(1,25)=1.0,2.0,4HDOUT,5.0,
2.0,25.00,0.,1.050,4.0,
KONFIG(1,26)=4HCNTL,SPCNTL(1,26)=1.0,3.0,4HDOUT,5.0,
3.0,25.00,0.,1.050,4.0,
KONFIG(1,27)=4HCNTL,SPCNTL(1,27)=1.0,5.0,4HSTAP,8.0,
9.0,0.,1.0,1.050,4.0,
KONFIG(1,29)=4HCNTL,SPCNTL(1,29)=1.0,1.0,4HSTAP,8.0,
3.0,0.,1.0,0.,0.,
KONFIG(1,33)=4HCNTL,SPCNTL(1,33)=1.0,18.00,4HSTAP,8.0,
13.00,0.,0.,1.0,0.,
KONFIG(1,34)=4HCNTL,SPCNTL(1,34)=1.0,9.0,4HDOUT,8.0,
18.00,0.,1.0,0.,0.,
KONFIG(1,35)=4HCNTL,SPCNTL(1,35)=1.0,4.0,4HSTAP,8.0,
7.0,0.,1.0,0.,0.,
KONFIG(1,37)=4HLIMV,0,0,0,0,SPEC(1,37)=0,0,8800.,4HDOUT,2.,18.,0,0,1.,
KONFIG(1,39)=4HCNTL,SPCNTL(1,39)=10.00,3.0,4HSTAP,8.0,
20.00,0.,0.,0.,40.00,
KONFIG(1,40)=4HLIMV,0,0,0,0,SPEC(1,40)=0,0,1.100,4HDOUT,6.,2.,0,0,1.,
KONFIG(1,45)=4HLIMV,0,0,0,0,SPEC(1,45)=0,0,1.100,4HDOUT,6.,3.,0,0,1.,
KONFIG(1,48)=4HLIMV,0,0,0,0,SPEC(1,48)=0,1.050,4.0,4HDOUT,4.,5.,0,0,1.,

```

```

KONFIG(1,49)=4HLIMV,0,0,0,0,SPEC(1,49)=0,0,1.100,4HDOUT,6.,5.,0,0,1.,
KONFIG(1,50)=4HLIMV,0,0,0,0,SPEC(1,50)=0,2.0,80.0,4HDOUT,5.,5.,0,0,1.,
KONFIG(1,51)=4HCNTL,SPCNTL(1,51)=10.00,2.0,4HSTAP,8.0,
16.00,0.,0.,0.,40.00,
KONFIG(1,52)=4HCNTL,SPCNTL(1,52)=1.0,7.0,4HSTAP,8.0,
5.0,0.,1.0,0.,40.00,
KONFIG(1,54)=4HOPTV,0,0,15,0,SPEC(1,54)=0.,0.,0.,1.0,0.,
0.,0.,0.,1.0,
KONFIG(1,55)=4HOPTV,0,0,20,0,SPEC(1,55)=0.,0.,0.,1.0,0.,
0.,0.,0.,1.0,
KONFIG(1,56)=4HOPTV,0,0,13,0,SPEC(1,56)=0.,0.,0.,1.0,0.,
0.,0.,0.,1.0,
KONFIG(1,58)=4HCNTL,SPCNTL(1,58)=4.0,6.0,4HDOUT,6.0,
3.0,1.0,0.,0.,0.,
KONFIG(1,59)=4HDUCT,4,0,5,0,SPEC(1,59)=0.,
&END
&D MACH=0.30,ALTP=5000.,ETAR=1.00,MODE=1, &END
&D MACH=0.50,ALTP=10000.,ETAR=1.00,MODE=1, &END
&D MACH=0.80,ALTP=20000.,ETAR=1.00,MODE=1, &END
&D MACH=1.00,ALTP=30000.,ETAR=1.00,MODE=1, &END
&D MACH=1.40,ALTP=40000.,ETAR=0.9782,MODE=1, &END
&D MACH=1.80,ALTP=50000.,ETAR=0.9445,MODE=1, &END
&D MACH=0.00,ALTP=0.,ETAR=1.00,MODE=2,
XNUM=4HINTI,4HSPLI,4HFFNI,4HVLVI,4H2FNI,4H2FNO,4HHPCI,4HHPCO,4HHPTI,
4HLPTI,4HLPTO,4H ,4HNOZI,4HNOZO,4HFFNO,4HDNZI,4HDNZO,4HCOOL,4HBDUC,
4HBNZI,4HBNZO,
SPEC(4,6)=2860,
SPEC(7,15)=0,SPEC(9,51)=0,SPEC(9,39)=0,IDONE(7)=0, &END
&D MACH=0.30,ALTP=5000.,ETAR=1.00,MODE=2,
SPEC(9,51)=1,SPEC(9,39)=1,SPEC(9,33)=1, &END
&D MACH=0.50,ALTP=10000.,ETAR=1.00,MODE=2, &END
&D MACH=0.80,ALTP=20000.,ETAR=1.00,MODE=2, &END
&D MACH=1.00,ALTP=30000.,ETAR=1.00,MODE=2, &END
&D MACH=1.40,ALTP=40000.,ETAR=0.9782,MODE=2, &END
&D MACH=1.80,ALTP=50000.,ETAR=0.9445,MODE=2, &END

```

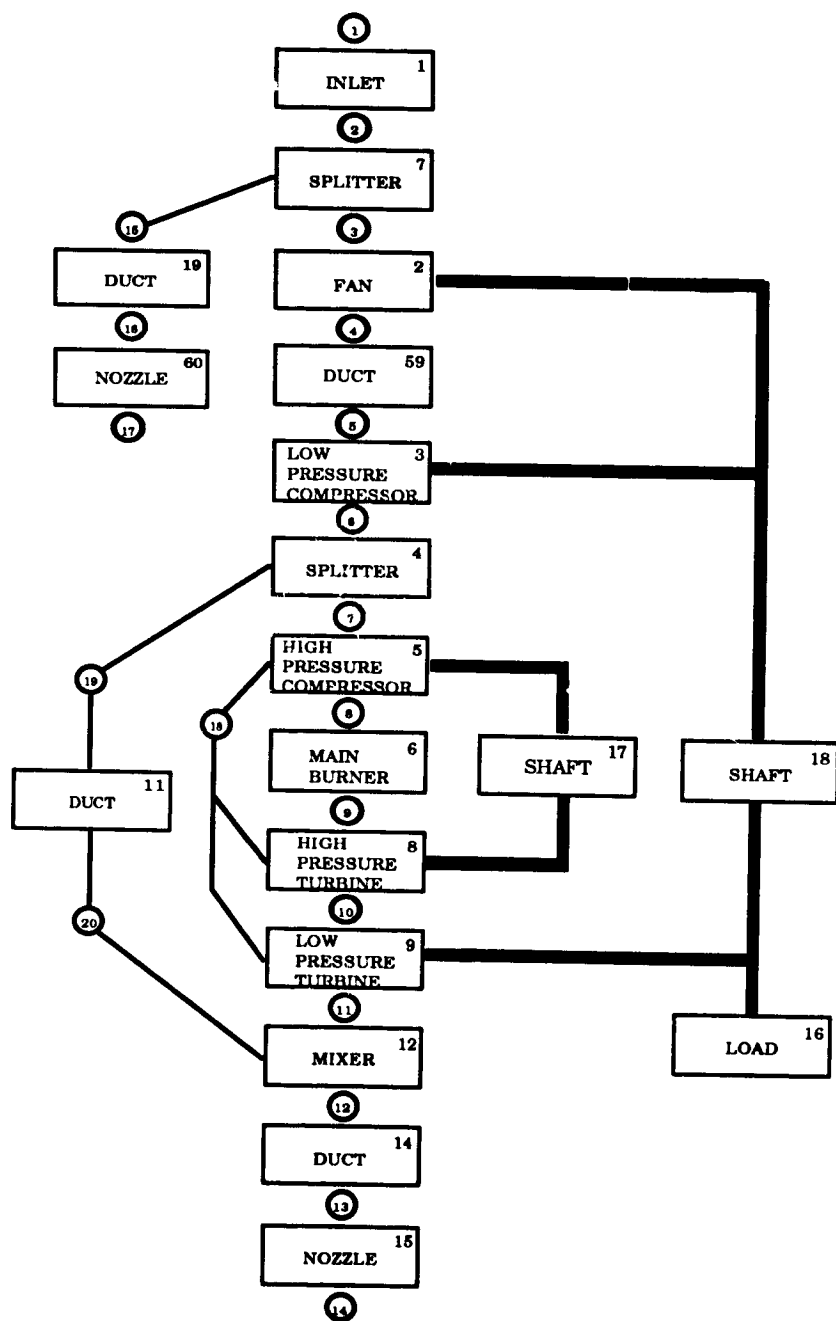


Figure 14. Block Diagram of Tandem Fan Engine, Mode 1, In Series, Mixed-Flow

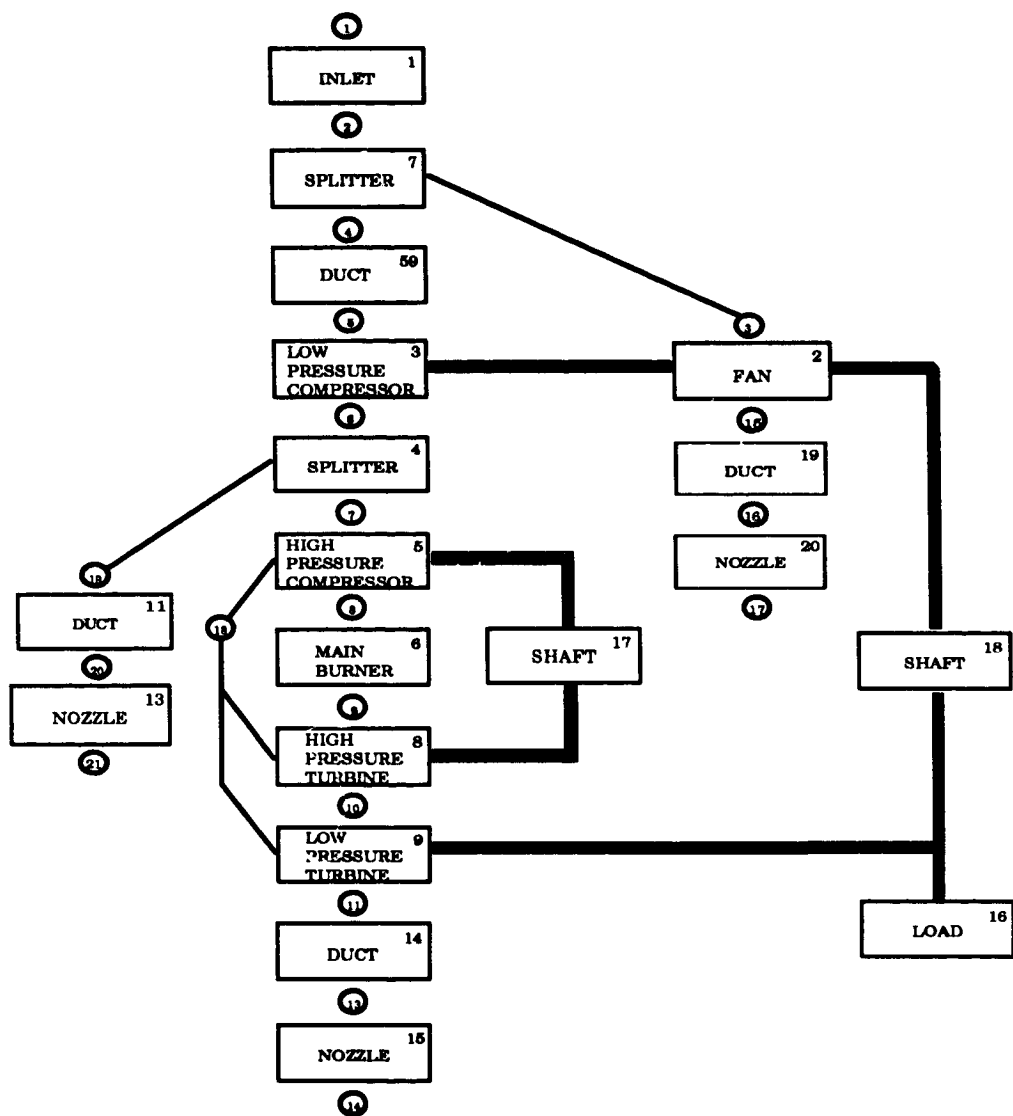


Figure 15. Block Diagram of Tandem Fan Engine, Mode 2, Parallel, High Bypass

4

The following is a sample output from the NNEP89 program. This output is from Sample 3, a simple mixed-flow, 1 spool turbofan (Section 12.3).

TEST CASE: MIXED FLOW TURBOFAN SANFORD NEW

6D LONG=T, NCODE=-1, DOUTH=-1, DRAW=T, 6END

1

TABLE DATA INPUT SUMMARY 10 TABLES

TABLE NUMBER	REFERENCE NUMBER	ARRAY LOCATION
--------------	------------------	----------------

1	1001	1
2	1002	1075
3	1003	2149
4	1004	3223
5	1005	4459
6	1006	5695
7	1007	6931
8	1008	7384
9	1009	7978
10	1010	8431

DATA STORAGE ALLOCATION	20000
DATA STORAGE NOT USED	10828

```

SD MODE=1,
XNM0=4HINLT,4HLPCT,4HSPLI,4HPCT,4HMERN,4HPTI,4HCRI,4HABRN,
XNM02I,4HNO2O,4HSECI,4HSECO,
CONFIG(1,1)=4HINLT,1,0,2,0,SPEC(1,1)=100.0,0.0,0.0,0.0,0.0,
0.0,0.0,0.18,00,
CONFIG(1,2)=4HCOMP,2,0,3,0,SPEC(1,2)=1.60,0.1,1001,1,1002,1,
1003,1,0.0,0.84,2,9,0.95,
CONFIG(1,3)=4HSPLT,3,0,4,12,SPEC(1,3)=1.75,0.02,0.02,
CONFIG(1,4)=4HCOMP,4,0,5,0,SPEC(1,4)=1.30,0.1,1004,1,1005,1,
1006,1,0.0,0.84,6,0,1,
CONFIG(1,5)=4HDUCT,5,0,6,0,SPEC(1,5)=.05,0.0,3000.,.99,18300.,
FARRAY(1,5)=2,1,
CONFIG(1,6)=4HTURB,6,0,7,0,SPEC(1,6)=3.6,1.0,1,1007,1,1008,
1,1,1,1,90,5000,1,
CONFIG(1,7)=4HMXR,7,13,8,0,SPEC(1,7)=0.0,0.0,40,1.,0.0.,
CONFIG(1,8)=4HDUCT,8,0,9,0,SPEC(1,8)=.05,0.0,3600.,.99,18300.,
FARRAY(1,8)=2,1,
CONFIG(1,9)=4HNOZZ,9,0,10,0,SPEC(1,9)=0.0,0.98,0.0.,.98,1.,1.,
0.1.,
CONFIG(1,10)=4HDUCT,12,0,13,0,SPEC(1,10)=.05,0.0,0.0,0.99,18300
FARRAY(1,10)=2,1,
CONFIG(1,11)=4HSEFT,2,4,6,0,SPEC(1,11)=9*1.0,
CONFIG(1,12)=4HCNTL,SPCNTL(1,12)=1,1,4HSTAP,8,2,0,0,
CONFIG(1,13)=4HCNTL,SPCNTL(1,13)=1,4,4HSTAP,8,4,0,0,
CONFIG(1,14)=4HCNTL,SPCNTL(1,14)=1,11,4HDOUT,8,11,0,0,
CONFIG(1,15)=4HCNTL,SPCNTL(1,15)=1,6,4HSTAP,8,6,0,0,
CONFIG(1,16)=4HCNTL,SPCNTL(1,16)=1,3,4HDOUT,8,7,0,0,
CONFIG(1,17)=4HCNTL,SPCNTL(1,17)=1,2,4HDOUT,5,2,50,1,
END

```

1	1>	
<INLT	2	
2	2>	
<COMP	3	
3	3>	<SPLT 3>



4  
 <COMP 4> 12  
 5 <DUCT 10> 13  
 6 <MIXR 7>  
 7  
 <TURB 6>  
 8  
 <MIXR 7>  
 9  
 <DUCT 8>  
 10  
 <NOZZ 9>

OSHAFT (11) IS CONNECTED TO COMP ( 2) AND COMP ( 4) AND TURB ( 6) AND  
 0 THE FOLLOWING REPRESENTS THE CONFIGURATION FOR MODE= 1  
 TEST CASE: MIXED FLOW TURBOFAN SANFORD NEW  
 CONFIGURATION DATA 13 STATIONS 17 COMPONENTS

COMPONENT NUMBER	NKIND	COMPONENT TYPE	UPSTREAM STATIONS	DOWNSTREAM STATIONS
1	1	INLET	1	0
2	4	COMPRES	2	0
3	7	SPLITTER	3	0
4	4	COMPRES	4	0
5	2	DUCT B	5	0
6	5	TURBINE	6	0
7	8	MIXER	7	0
8	2	DUCT B	8	0
9	9	NOZZLE	9	0
10	2	DUCT B	12	0
11	11	SHAFT	2	4
12	12	CONTROL	2	0
13	12	CONTROL	4	0
14	12	CONTROL	11	0
15	12	CONTROL	6	0
16	12	CONTROL	7	0
17	12	CONTROL	2	0

# CONTROL INFORMATION

12 VARY DATINP 1 OF COMPONENT 1 SO THAT STATP 8 OF FLOW STATION 2 EQUALS 0.00000D+00  
 13 VARY DATINP 1 OF COMPONENT 4 SO THAT STATP 8 OF FLOW STATION 4 EQUALS 0.00000D+00  
 14 VARY DATINP 1 OF COMPONENT 11 SO THAT DATOUT 8 OF COMPONENT 11 EQUALS 0.00000D+00  
 15 VARY DATINP 1 OF COMPONENT 6 SO THAT STATP 8 OF FLOW STATION 6 EQUALS 0.00000D+00  
 16 VARY DATINP 1 OF COMPONENT 3 SO THAT DATOUT 8 OF COMPONENT 7 EQUALS 0.00000D+00  
 17 VARY DATINP 1 OF COMPONENT 2 SO THAT DATOUT 5 OF COMPONENT 2 EQUALS 0.50000D+02  
 OCASE IDENTIFICATION TEST CASE: MIXED FLOW TURBOFAN SANFORD NEW



STATIC PRESSURE IN PRIMARY > TOTAL PRESSURE IN SECONDARY  
IGNORE THIS MESSAGE IF IT ONLY OCCURS ONCE

# STATION PROPERTY OUTPUT DATA

FLOW STATION	WEIGHT FLOW	TOTAL PRESSURE	TOTAL TEMPERATURE	FUEL/AIR RATIO	CORRECTED FLOW	MACH NUMBER	STATIC PRESSURE	INTERFACE FLOW ERROR
WHEP-ARP?	STATP1	STATP2	STATP3	STATP4	STATP5	STATP6	STATP7	STATP8
1-INLET	100.0000000	14.69600010	536.6700550	0.000000000	101.7184531	0.000000000	0.000000000	0.000000000
2-LPCI	100.0000000	14.69600010	536.6700550	0.000000000	101.7184531	0.000000000	0.000000000	0.000000000
3-SPLI	100.0000000	42.61840029	762.9536480	0.000000000	41.82126301	0.000000000	0.000000000	0.000000000
4-HECI	36.36363636	41.76603228	762.9536480	0.000000000	15.51809388	0.000000000	0.000000000	0.000000000
5-BERN	36.36363636	250.5961937	1340.452597	0.000000000	3.428181271	0.000000000	0.000000000	0.000000000
6-HPTI	37.39190933	238.0663840	3000.000000	0.028277507	5.551181223	0.000000000	0.000000000	0.000000000
7-MERI	37.39190933	35.37885436	2048.875601	0.028277507	30.87001806	0.400000000	31.89916373	0.000000000
8-ABRN	101.0282730	36.74156259	1274.420012	0.010282730	63.34127621	0.500679729	31.11792949	0.000000000
9-NOZI	105.3013724	34.90448446	3600.000000	0.053013724	116.8017274	1.000000000	19.24826153	0.000000000
10-NOZO	105.3013724	34.90448446	3600.000000	0.053013724	116.8017274	1.197803145	14.69600010	0.000000000
12-SECO	63.63636364	41.76603228	762.9536480	0.000000000	27.15666429	0.000000000	0.000000000	0.000000000
13-13	63.63636364	39.67773067	762.9536480	0.000000000	28.58596242	0.567691505	31.89916373	0.000000000

## COMPONENT OUTPUT DATA

COMPONENT NO. TYPE	DATOUT1	DATOUT2	DATOUT3	DATOUT4	DATOUT5	DATOUT6	DATOUT7	DATOUT8	DATOUT9
1 INLET	DRAG	V-FPS	V-KNT	T2/T1	P2/P1	MACH	ETAR	T2/518.67	ALT-FT
COMPRES	HP IN	RPM ACT	Z 3DMAP	R MAP	% MARGIN	CORR. SPD	FLOW SCALE	ETA COMP	PR COMP
2 COMPRES	-7717.837523	1.000000000	0.000000000	2.474319065	49.99992041	0.950000000	103.4959524	0.840000000	2.900000000
SPLITTER	BYPASS	DE/P MAIN	DE/P 2ND						
3 SPLITTER	1.750000000	0.020000000	0.020000000						
COMPRES	HP IN	RPM ACT	Z 3DMAP	R MAP	% MARGIN	CORR. SPD	FLOW SCALE	ETA COMP	PR COMP
4 COMPRES	-7440.631442	1.000000000	0.000000000	1.300000000	28.12980162	1.000000000	15.14087666	0.840000000	6.000000000
DUCT B	DE/P MOM	DE/P	PTOT/EST IN	FUEL/WA	A-SQ IN	FUEL FLOW	MACH IN	ETA BURN	T BURN
5 DUCT B	0.000000000	0.050000000	0.000000000	0.028277507	0.000000000	3701.782682	0.000000000	0.990000000	3000.000000
TURBINE	HP OUT	RPM ACT	Z 3DMAP	PR-TABLE	SPD. SCALE	CORR. SPD	FLOW SCALE	ETA TURB	PR TURB
6 TURBINE	15158.46896	1.000000000	1.000000000	3.600000000	0.000083160	5000.000000	0.281791278	0.900000000	6.729058594
MIXER	MAIN-SQIN	A2ND-SQIN	PT/PS-MAIN	PT/PS 2ND	V MAIN	V 2ND	DELSTAT	AXIT/AENT	
7 MIXER	147.4645721	102.5900394	1.109084071	1.243848616	847.6687865	744.0829814	844.6354467	0.000000000	1.000000000
DUCT B	DE/P MOM	DE/P	PTOT/EST IN	FUEL/WA	A-SQ IN	FUEL FLOW	MACH IN	ETA BURN	T BURN
8 DUCT B	0.000000000	0.050000000	0.000000000	0.042296075	0.000000000	15383.15794	0.000000000	0.990000000	3600.000000
NOZZLE	F GROSS	VJ ACT	PTOT/PS	AX-SQIN	ATRT-SQIN	CD, FLOW	CV, VEL	PR CRITICAL	PR NOZZ
9 NOZZLE	10025.67366	3063.265152	2.375100995	374.0212322	359.8791617	0.980000000	0.980000000	1.813383739	2.375100995
DUCT B	DE/P MOM	DE/P	PTOT/EST IN	FUEL/WA	A-SQ IN	FUEL FLOW	MACH IN	ETA BURN	T BURN
10 DUCT B	0.000000000	0.050000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	762.9536480
SHAFT	NET HP	RPM ACT	RPM UP1	RPM UP2	RPM DN1	RPM DN2	NETHP/TOTHP		
11 SHAFT	0.000000000	1.000000000	1.000000000	1.000000000	1.000000000	0.000000000	0.000000000	0.000000000	0.000000000

## CONTROL INFORMATION

17

MACH= 0.0000 ALTITUDE= 0. RECOVERY= 1.0000 8 ITERATIONS 20 PASSES

AIRFLOW (LB/SEC)	100.00	GROSS THRUST	10025.67	FUEL FLOW (LB/HR)	19084.94
NET THRUST	10025.67	TSFC	1.9036	NET THRUST/AIRFLOW	100.2567
TOTAL INLET DRAG	0.00	TOTAL BRAKE SHAFT HP	0.00	BOATTAILED DRAG	0.00
INSTALLED THRUST	10025.67	INSTALLED TSFC	1.9036	SPILLAGE + LIP DRAG	0.00

1 4D SPEC(9,12)=1,SPEC(9,13)=1,SPEC(9,14)=1,SPEC(9,15)=1,  
SPEC(5,1)=26,SPEC(9,1)=50,SPEC(6,1)=0,  
SPEC(9,16)=1,SPEC(9,17)=1 6END

ONODE 1 NOW BEING USED  
SELAST 0.33584D-04 SE 0.10505D+00 SCM 0.10000D+01 SRATIO 0.51169D-04 XMAX 0.27349D-04  
SELAST 0.10505D+00 SE 0.13302D-02 SCM 0.10000D+01 SRATIO 0.51169D-04 XMAX 0.30041D-01  
SELAST 0.13302D-02 SE 0.23202D-05 SCM 0.10000D+01 SRATIO 0.12663D-01 XMAX 0.23690D-03  
SELAST 0.23202D-05 SE 0.48337D-10 SCM 0.10000D+01 SRATIO 0.17443D-02 XMAX 0.82845D-05  
1CASE IDENTIFICATION TEST CASE: MIXED FLOW TURBOFAN SANFORD NEW

# STATION PROPERTY OUTPUT DATA

FLOW STATION NAME-APP?	WEIGHT FLOW STATP1	TOTAL PRESSURE STATP2	TOTAL TEMPERATURE STATP3	FUEL/AIR RATIO STATP4	CORRECTED FLOW STATP5	MACH NUMBER STATP6	STATIC INTERFACE CORRECTED	
							PRESSURE STATP7	FLOW ERROR STATP8
1-INLET	103.0290200	14.66946595	536.4917474	0.000000000	104.9716443	0.260000000	0.000000000	0.000000000
2-LPCI	103.0290437	15.37708409	543.7477516	0.000000000	100.8160153	0.000000000	0.000000000	-0.23060D-06
3-SPLI	103.0290437	43.85269181	768.3528973	0.000000000	42.02318745	0.000000000	0.000000000	0.000000000
4-RECI	36.97811271	42.97563798	768.3528973	0.000000000	15.39033218	0.000000000	0.000000000	0.13656D-07
5-MERN	36.97811271	254.7288306	1340.148248	0.000000000	3.429164138	0.000000000	0.000000000	0.000000000
6-HPTI	38.0239380	241.9923890	3000.000000	0.028282322	5.553429249	0.000700000	0.000000000	-0.54421D-08
7-MERI	38.0239380	36.09679794	2050.527695	0.028282322	30.77984771	0.398390952	32.58338021	0.000000000
8-ABRN	104.0748703	37.66093711	1271.994846	0.010150797	63.59788069	0.503449957	31.83830870	0.000000000
9-NOZI	108.4799590	35.77789025	3600.000000	0.052906592	117.3900390	1.000000000	19.72950497	0.000000000
10-NOZO	108.4799590	35.77789025	3600.000000	0.052906592	117.3900390	1.217668300	14.66946595	0.000000000
12-SECO	66.05093053	42.97563798	768.3528973	0.000000000	27.49047134	0.000000000	0.000000000	0.000000000
13-13	66.05093053	40.82685608	768.3528973	0.000000000	28.93733825	0.578275401	32.583337373	0.000000000

# COMPONENT OUTPUT DATA

COMPONENT NO. TYPE	DATOUT1	DATOUT2	DATOUT3	DATOUT4	DATOUT5	DATOUT6	DATOUT7	DATOUT8	DATOUT9
1 INLET	DRAG	V-FPS	V-KNT	T2/T1	P2/P1	MACH	ETAR	T2/518.67	ALT-FT
2 COMPRESR	HP IN	RPM ACT	Z 3DMAP	R MAP	% MARGIN	CORR. SPD	FLOW SCALE	ETA COMP	PR COMP
3 SPLITTER	BYPASS	DP/P MAIN	DP/P 2ND	DP/P 2ND	DP/P 2ND	0.939581044	103.4959524	0.841603652	2.851821031
4 COMPRESR	HP IN	RPM ACT	Z 3DMAP	R MAP	% MARGIN	0.000000000	0.000000000	0.000000000	0.000000000
5 DUCT B	DP/P MOM	DP/P	PTOT/PT IN	FUEL/WA	A-SQ. IN	FUEL FLOW	MACH IN	ETA BURN	T BURN
6 TURBINE	HP OUT	RPM ACT	Z 3DMAP	PR-TABLE	SPD. SCALE	CORR. SPD	FLOW SCALE	ETA TURB	PR TURB
7 MIXER	AMAIN-SQIN	A2ND-SQIN	PT/PS-MAIN	PT/PS 2ND	V MAIN	V 2ND	V EXIT	DELSTAT	AEXIT/AENT
8 DUCT B	DP/P MOM	DP/P	PTOT/PT IN	FUEL/WA	A-SQ. IN	FUEL FLOW	MACH IN	ETA BURN	T BURN
9 NOZZLE	F GROSS	VJ ACT	PTOT/PS	AEX-SQIN	ATHRT-SQIN	CD, FLOW	CV, VEL	PR CRITICAL	PR NOZZ
10 DUCT B	DP/P MOM	DP/P	PTOT/PT IN	FUEL/WA	A-SQ. IN	FUEL FLOW	MACH IN	ETA BURN	T BURN
11 SHAFT	NET HP	RPM ACT	RPM UP1	RPM UP2	RPM DN1	RPM DN2	0.000000000	0.000000000	0.000000000
12 SHAFT	0.001502450	0.995533082	0.995533082	0.995533082	0.995533082	0.000000000	0.000000000	0.97632D-07	0.000000000

ACTIVE CONTROLS:	15	13	16	17	12	14	CONTROL INFORMATION
MACH- 0.2600	ALTITUDE-	50.	RECOVERY-	1.0000	3	ITERATIONS	10 PASSES
AIRFLOW (LB/SEC)		103.03	GROSS THRUST		10471.21	FUEL FLOW (LB/HR)	
NET THRUST		9525.84	TSFC		2.0600	NET THRUST/AIRFLOW	19623.30
TOTAL INLET DRAG		945.37	TOTAL BRAKE SHAFT HP		0.00	BOAT TAIL DRAG	92.4578
INSTALLED THRUST		9525.84	INSTALLED TSFC		2.0600	SPILLAGE + LIP DRAG	0.00
1							0.00

## 12.7 EXAMPLE COMPRESSOR MAP TABLES

The following is a sample compressor map table with variable stator angles. Two stator angles (0.0, 10.0) are given in the table. The corresponding output graph of this table which results from setting MAPLOT=T is shown in Figure 16. The corrected speed values are labeled along the top of the graph and the R values are labeled on the right side of the graph. The efficiency contours are numbered and the corresponding values are listed in the lower right hand corner.

2001 HIGH PRESSURE COMPRESSOR WITH VARIABLE STATORS								
ANGL	2	0.00	10.00					
SPED	15	0.600	0.700	0.750	0.800	0.810	0.820	0.830
		0.840	0.850	0.860	0.870	0.900	0.935	0.985
		1.035						
R	7	1.000	1.050	1.150	1.300	1.450	1.600	1.750
FLOW	7	0.3520	0.3580	0.3640	0.3730	0.3820	0.3840	0.3840
FLOW	7	0.3910	0.3960	0.4060	0.4140	0.4210	0.4250	0.4260
FLOW	7	0.4330	0.4440	0.4500	0.4550	0.4580	0.4600	0.4610
FLOW	7	0.4690	0.4840	0.4930	0.5000	0.5040	0.5050	0.5060
FLOW	7	0.5080	0.5230	0.5350	0.5480	0.5530	0.5550	0.5560
FLOW	7	0.5690	0.5820	0.5930	0.6080	0.6170	0.6210	0.6240
FLOW	7	0.6240	0.6380	0.6540	0.6700	0.6770	0.6800	0.6810
FLOW	7	0.6580	0.6720	0.6860	0.7020	0.7100	0.7140	0.7160
FLOW	7	0.6880	0.7030	0.7160	0.7300	0.7370	0.7410	0.7440
FLOW	7	0.7240	0.7350	0.7510	0.7660	0.7750	0.7780	0.7800
FLOW	7	0.7580	0.7710	0.7850	0.8020	0.8090	0.8110	0.8150
FLOW	7	0.8430	0.8550	0.8680	0.8800	0.8850	0.8890	0.8910
FLOW	7	0.9120	0.9250	0.9350	0.9480	0.9510	0.9530	0.9540
FLOW	7	0.9860	0.9950	1.0040	1.0090	1.0100	1.0100	1.0100
FLOW	7	1.0600	1.0600	1.0600	1.0600	1.0600	1.0600	1.0600
SPED	15	0.600	0.700	0.750	0.800	0.810	0.820	0.830
		0.840	0.850	0.860	0.870	0.900	0.935	0.985
		1.035						
R	7	1.000	1.050	1.150	1.300	1.450	1.600	1.750
FLOW	7	0.7520	0.7580	0.7640	0.7730	0.7820	0.7840	0.7840
FLOW	7	0.7910	0.7960	0.8060	0.8140	0.8210	0.8250	0.8260
FLOW	7	0.8330	0.8440	0.8500	0.8550	0.8580	0.8600	0.8610
FLOW	7	0.8690	0.8840	0.8930	0.9000	0.9040	0.9050	0.9060
FLOW	7	0.9080	0.9230	0.9350	0.9480	0.9530	0.9550	0.9560
FLOW	7	0.9690	0.9820	0.9930	1.0080	1.0170	1.0210	1.0240
FLOW	7	1.0240	1.0380	1.0540	1.0700	1.0770	1.0800	1.0810
FLOW	7	1.0580	1.0720	1.0860	1.1020	1.1100	1.1140	1.1160
FLOW	7	1.0880	1.1030	1.1160	1.1300	1.1370	1.1410	1.1440
FLOW	7	1.1240	1.1350	1.1510	1.1660	1.1750	1.1780	1.1800
FLOW	7	1.1580	1.1710	1.1850	1.2020	1.2090	1.2110	1.2150
FLOW	7	1.2430	1.2550	1.2680	1.2800	1.2850	1.2890	1.2910
FLOW	7	1.3120	1.3250	1.3350	1.3480	1.3510	1.3530	1.3540
FLOW	7	1.3860	1.3950	1.4040	1.4090	1.4100	1.4100	1.4100
FLOW	7	1.4600	1.4600	1.4600	1.4600	1.4600	1.4600	1.4600
EOT								
2002 HIGH PRESSURE COMPRESSOR WITH VARIABLE STATORS								
ANGL	2	0.00	10.00					
SPED	15	0.600	0.700	0.750	0.800	0.810	0.820	0.830
		0.840	0.850	0.860	0.870	0.900	0.935	0.985
		1.035						
R	7	1.000	1.050	1.150	1.300	1.450	1.600	1.750
EFF	7	0.9000	0.8500	0.7160	0.5450	0.3400	0.2000	0.2000
EFF	7	0.9400	0.9080	0.8350	0.6600	0.4450	0.3150	0.2800
EFF	7	0.9540	0.9400	0.8830	0.7650	0.5350	0.2850	0.1800
EFF	7	0.9640	0.9540	0.9160	0.8120	0.6450	0.4000	0.2400
EFF	7	0.9730	0.9660	0.9430	0.8650	0.7450	0.5620	0.3200
EFF	7	0.9860	0.9790	0.9650	0.9050	0.8180	0.6950	0.5470
EFF	7	0.9960	0.9920	0.9820	0.9380	0.8680	0.7660	0.6350
EFF	7	1.0030	0.9980	0.9900	0.9500	0.8900	0.7890	0.6660
EFF	7	1.0070	1.0040	0.9970	0.9520	0.9030	0.8070	0.6810
EFF	7	1.0110	1.0090	1.0030	0.9710	0.9180	0.8300	0.7040

EFF	7	1.0140	1.0120	1.0080	0.9830	0.9310	0.8360	0.7100
EFF	7	1.0180	1.0150	1.0150	0.9950	0.9420	0.8550	0.7250
FFF	7	1.0150	1.0140	1.0110	0.9820	0.9300	0.8430	0.7000
FFF	7	1.0070	1.0010	0.9930	0.9570	0.9060	0.8160	0.6680
EFF	7	0.9180	0.9180	0.9090	0.8900	0.8530	0.7740	0.6070
SPED	15	0.600	0.700	0.750	0.800	0.810	0.820	0.830
		0.840	0.850	0.860	0.870	0.900	0.935	0.985
		1.035						
R	7	1.000	1.050	1.150	1.300	1.450	1.600	1.750
EFF	7	0.8550	0.8075	0.6802	0.5177	0.3230	0.1900	0.1900
EFF	7	0.8930	0.8626	0.7932	0.6270	0.4227	0.2993	0.2660
EFF	7	0.9063	0.8930	0.8388	0.7268	0.5082	0.2707	0.1710
EFF	7	0.9158	0.9063	0.8702	0.7714	0.6127	0.3800	0.2280
EFF	7	0.9244	0.9177	0.8959	0.8217	0.7077	0.5339	0.3040
EFF	7	0.9367	0.9301	0.9168	0.8597	0.7771	0.6603	0.5196
EFF	7	0.9462	0.9424	0.9329	0.8911	0.8246	0.7277	0.6032
EFF	7	0.9528	0.9481	0.9405	0.9025	0.8455	0.7495	0.6327
EFF	7	0.9567	0.9538	0.9471	0.9139	0.8579	0.7666	0.6470
EFF	7	0.9604	0.9585	0.9528	0.9224	0.8721	0.7885	0.6588
EFF	7	0.9633	0.9614	0.9576	0.9338	0.8845	0.7942	0.6745
EFF	7	0.9671	0.9642	0.9642	0.9452	0.8949	0.8122	0.6887
EFF	7	0.9642	0.9633	0.9604	0.9329	0.8835	0.8009	0.6650
EFF	7	0.9567	0.9509	0.9434	0.9092	0.8607	0.7752	0.6346
EFF	7	0.8721	0.8721	0.8636	0.8455	0.8104	0.7353	0.5766
EOT								
0003								
ANGL	2	0.00	10.00					
SPED	15	0.600	0.700	0.750	0.800	0.810	0.820	0.830
		0.840	0.850	0.860	0.870	0.900	0.935	0.985
		1.035						
R	7	1.000	1.050	1.150	1.300	1.450	1.600	1.750
PR	7	2.0730	2.0120	1.7360	1.4600	1.1530	1.0000	1.0000
PR	7	2.5940	2.4410	2.1880	1.8280	1.4750	1.2840	1.2300
PR	7	3.1610	2.9470	2.6250	2.2260	1.7970	1.5360	1.4220
PR	7	3.6060	3.3910	3.0690	2.6090	2.1650	1.8430	1.6670
PR	7	4.0810	3.8200	3.5290	3.0690	2.5940	2.1880	1.9430
PR	7	4.7400	4.4570	4.1420	3.6210	3.1080	2.6480	2.3030
PR	7	5.3230	5.0700	4.7400	4.1960	3.6060	3.0690	2.6320
PR	7	5.6750	5.4300	5.0620	4.4870	3.8890	3.2990	2.8240
PR	7	6.0050	5.7670	5.3840	4.7550	4.1040	3.4910	2.9620
PR	7	6.4030	6.1350	5.7360	5.0770	4.4030	3.7590	3.1690
PR	7	6.8020	6.5030	6.0970	5.4070	4.6940	3.9660	3.3610
PR	7	7.8210	7.4230	6.9550	6.1350	5.3300	4.5100	3.7740
PR	7	8.6490	8.1810	7.6600	6.7480	5.8510	4.9700	4.1040
PR	7	9.3690	8.9250	8.3580	7.3460	6.3500	5.3530	4.4340
PR	7	9.8140	9.6530	8.9630	7.8130	6.7710	5.7060	4.7020
SPED	15	0.600	0.700	0.750	0.800	0.810	0.820	0.830
		0.840	0.850	0.860	0.870	0.900	0.935	0.985
		1.035						
R	7	1.000	1.050	1.150	1.300	1.450	1.600	1.750
PR	7	2.7919	2.6900	2.2291	1.7682	1.2555	1.0000	1.0000
PR	7	3.6620	3.4065	2.9840	2.3828	1.7932	1.4743	1.3841
PR	7	4.6089	4.2515	3.7137	3.0474	2.3310	1.8951	1.7047
PR	7	5.3520	4.9930	4.4552	3.6870	2.9455	2.4078	2.1139
PR	7	6.1453	5.7094	5.2234	4.4552	3.6620	2.9840	2.5748
PR	7	7.2458	6.7732	6.2471	5.3771	4.5204	3.7522	3.1760
PR	7	8.2194	7.7969	7.2458	6.3373	5.3520	4.4552	3.7254
PR	7	8.8072	8.3981	7.7835	6.8233	5.8246	4.8393	4.0461
PR	7	9.3583	8.9609	8.3213	7.2708	6.1837	5.1600	4.2765
PR	7	10.0230	9.5754	8.9091	7.8086	6.6830	5.6075	4.6222
PR	7	10.6893	10.1900	9.5120	8.3597	7.1690	5.9532	4.9429
PR	7	12.3911	11.7264	10.9448	9.5754	8.2311	6.8617	5.6326
PR	7	13.7738	12.9923	12.1222	10.5992	9.1012	7.6299	6.1837
PR	7	14.9762	14.2347	13.2879	11.5978	9.9345	8.2695	6.7348
PR	7	15.7194	15.4505	14.2982	12.3777	10.6376	8.8590	7.1823
EOT								

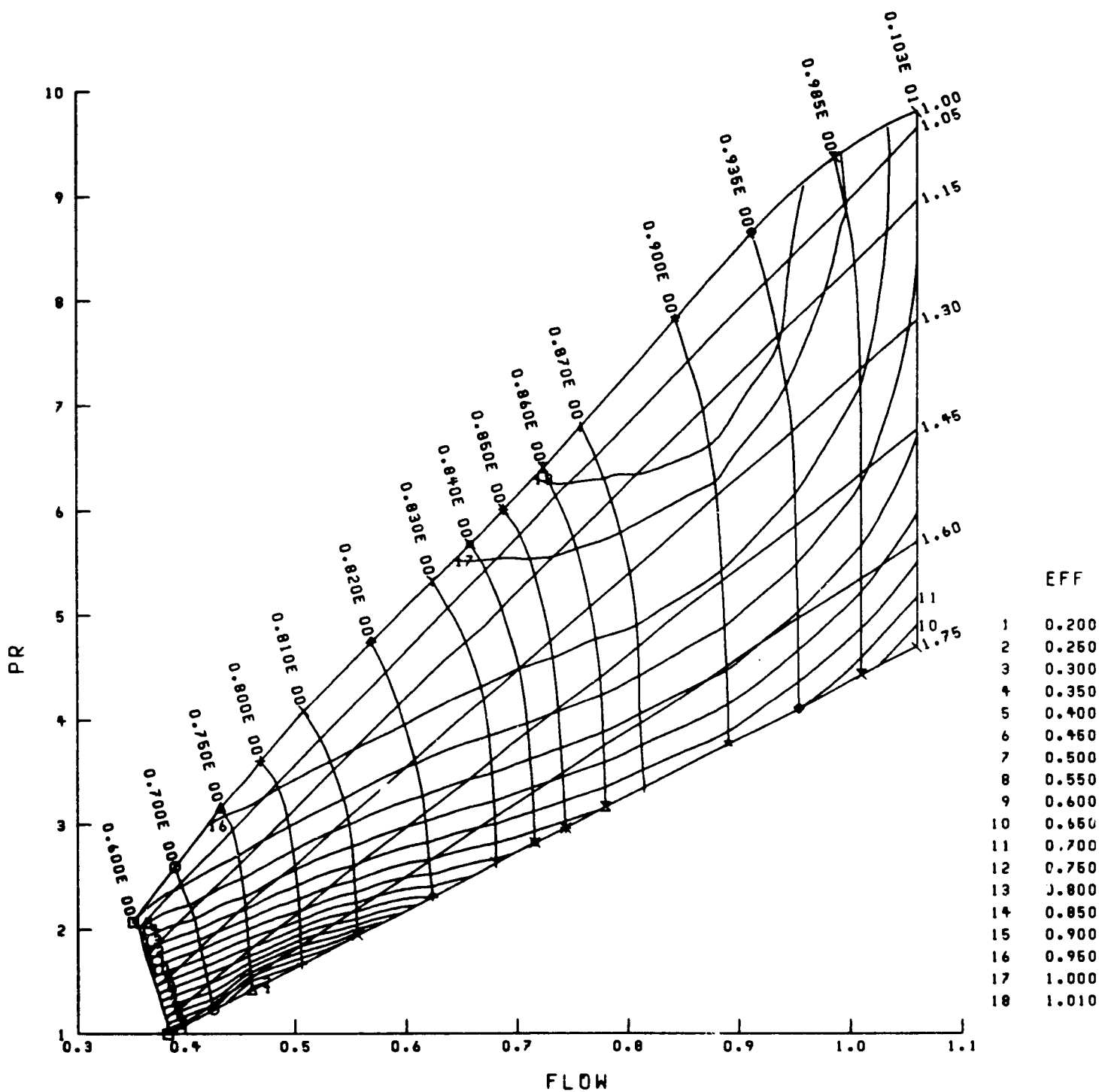


Figure 16. Example High Pressure Compressor Map with Stator Angle=0.



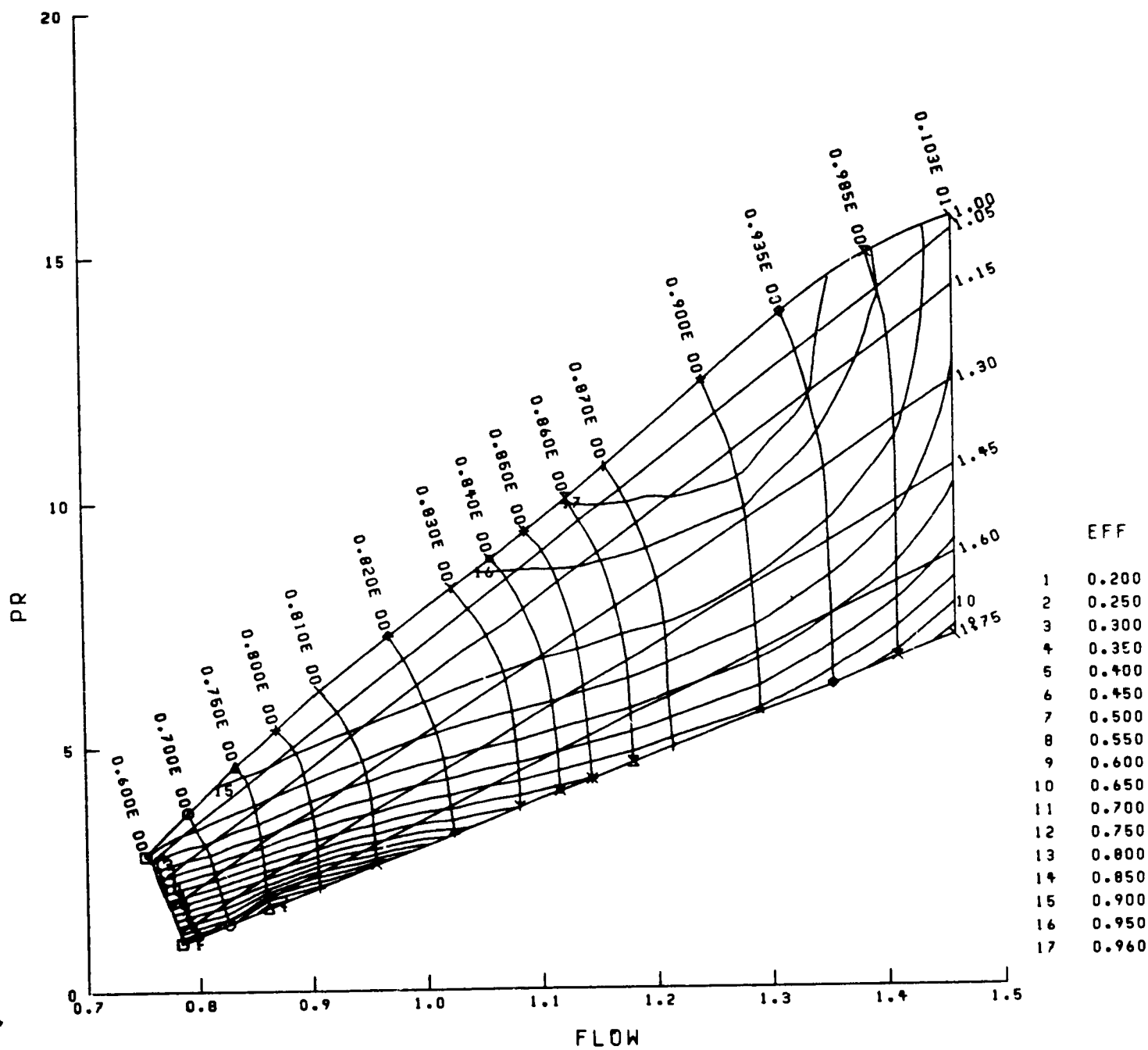


Figure 16 continued. Example High Pressure Compressor Map with Stator Angle=10.

## 12.8 EXAMPLE TURBINE MAP TABLES

The following is a sample turbine map table. The corresponding output graph of this table which results from setting MAPLOT=T is shown in Figure 17. The corrected speed values are labeled with symbols and the corresponding speed values are listed in the lower right hand corner of the graph.

2004		HIGH PRESSURE TURBINE							
AREA	1	1.00							
SPED	3	4000. 5000. 6000.							
PR	14	1.000	1.300	1.500	1.600	1.800	2.000	2.200	
		2.500	2.800	3.100	3.300	3.500	3.600	5.000	
FLOW	14	0.000	15.300	17.100	17.775	18.625	19.150	19.460	
		19.750	19.900	19.980	20.010	20.040	20.040	20.041	
FLOW	14	0.000	15.775	17.100	17.575	18.225	18.700	19.040	
		19.360	19.540	19.640	19.670	19.700	19.700	19.701	
FLOW	14	0.000	16.225	17.125	17.500	18.040	18.450	18.750	
		19.050	19.1'0	19.280	19.310	19.340	19.340	19.341	
EOT									
2005		HIGH PRESSURE TURBINE							
AREA	1	1.00							
SPED	4	4000.0 5000.0 6000.0 8000.0							
PR	14	1.000	1.250	1.750	2.000	2.150	2.380	2.500	
		2.750	3.250	3.500	4.000	4.500	4.750	5.000	
EFF	14	0.8370	0.8419	0.8512	0.8557	0.8581	0.8615	0.8635	
		0.8672	0.8734	0.8760	0.8786	0.8790	0.8782	0.8770	
EFF	14	0.8400	0.8495	0.8657	0.8725	0.8765	0.8806	0.8811	
		0.8815	0.8819	0.8820	0.8802	0.8762	0.8733	0.8700	
EFF	14	0.8400	0.8492	0.8648	0.8705	0.8735	0.8772	0.8792	
		0.8822	0.8867	0.8881	0.8891	0.8877	0.8862	0.8840	
EFF	14	0.8400	0.8489	0.8636	0.8687	0.8711	0.8726	0.8732	
		0.8736	0.8727	0.8720	0.8705	0.8688	0.8678	0.8668	
EOT									

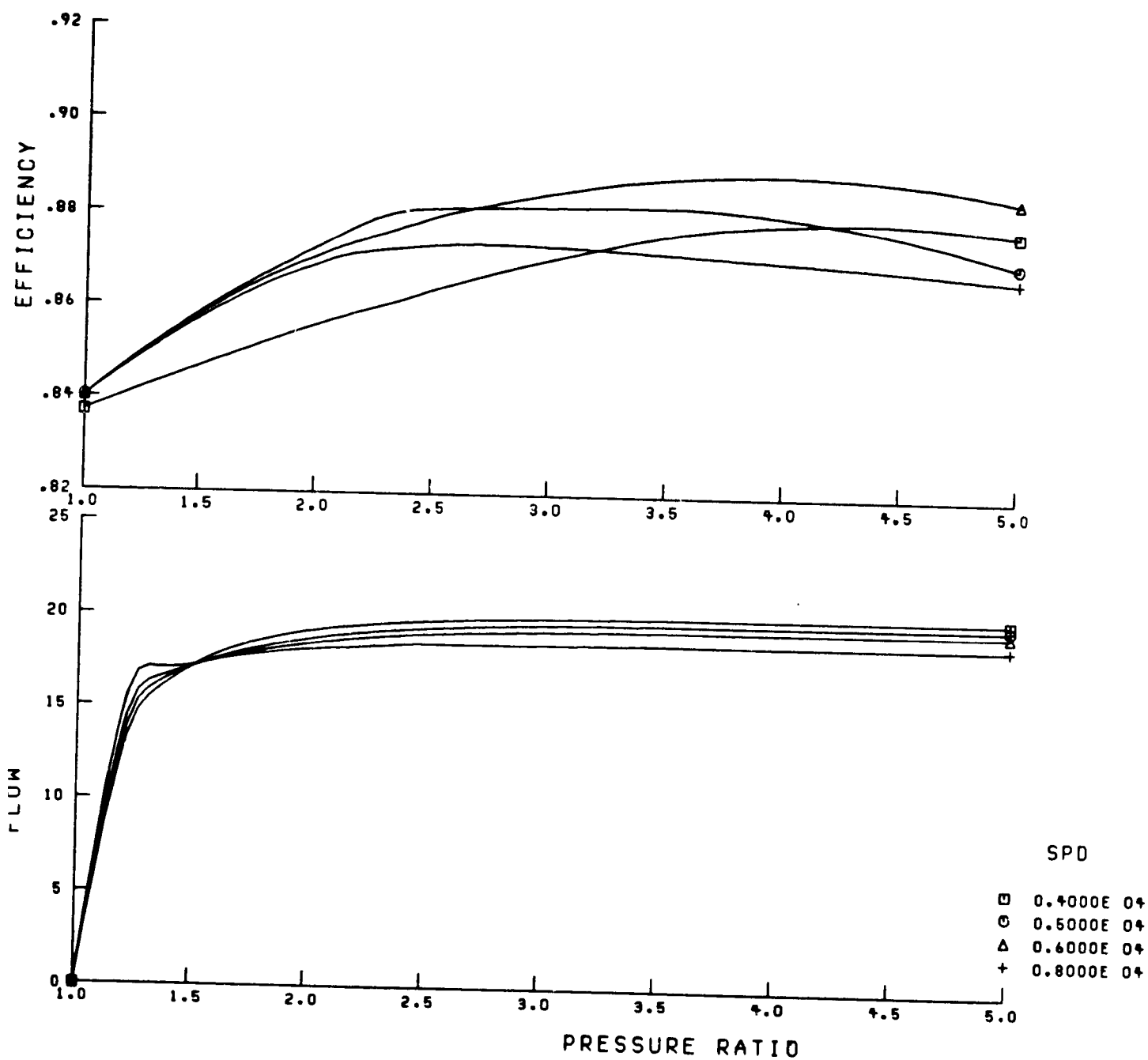


Figure 17. Example High Pressure Turbine Map

## ACKNOWLEDGEMENT

The authors wish to thank the members of the Aeropropulsion Analysis Office of the NASA Lewis Research Center. Their assistance, suggestions, and testing of the various versions of the program and this manual were invaluable.

## REFERENCES

1. Fishbach, Laurence H.; and Caddy, Michael J.: NNEP - The Navy NASA Engine Program. NASA TM-X-71857, 1975.
2. Caddy, Michael J.; and Shapiro, Stanley R.: NEPCOMP - The Navy Engine Performance Computer Program, Version I. NADC-74045-30, 1975.
3. Fishbach, Laurence H.; and Gordon, Sanford: NNEPEQ - Chemical Equilibrium Version of the Navy/NASA Engine Program. NASA TM-100851, 1988.
4. Gordon, Sanford: The Navy/NASA Engine Program (NNEP89) - Interfacing the Program for the Calculation of Complex Chemical Equilibrium Compositions (CEC). To be published.
5. Plencner, Robert M.; Senty, P.; and Wickenheiser, T.J.: Propeller Performance and Weight Prediction Appended to the Navy/NASA Engine Program. NASA TM-83458, 1983.
6. Plencner, Robert M.: Plotting Component Maps in the Navy/NASA Engine Program (NNEP)-A Method and Its Usage. NASA TM 101433, 1989.
7. Berton, Jeffrey J.: Divergence Thrust Loss Calculations for Convergent-Divergent Nozzles: Extensions to the Classical Case. To be published.
8. Plencner, Robert M.: Automatic Controls in the Navy/NASA Engine Program (NNEP) and Its Usage. To be published.
9. Gauntner, James W.: Algorithm for Calculating Turbine Cooling Flow and the Resulting Decrease in Turbine Efficiency. NASA TM-81453, 1980.
10. Onat, E.; and Klees, G. W.: A Method to Estimate Weight and Dimensions of Large and Small Gas Turbine Engines. NASA CR-159481, Jan. 1979.
11. Kowalski, Edward J.; and Atkins, Robert A., Jr.: Computer Code for Estimating Installed Performance of Aircraft Gas Turbine Engines, Volume II User's Manual. NASA CR-159692, Dec. 1979.
12. Converse, G. L.; and Giffin, R. G.: Extended Parametric Representation of Compressor, Fans and Turbines Volume I - CMGEN User's Manual. NASA CR-174645, March 1984.
13. Converse, G. L.: Extended Parametric Representation of Compressor, Fans and Turbines Volume III - MODFAN User's Manual (Parametric Modulating Flow Fan). NASA CR-174647, March 1984.
14. Converse, G. L.: Extended Parametric Representation of Compressor, Fans and Turbines Volume II - PART User's Manual (Parametric Turbine). NASA CR-174646, March 1984.

15. Tran, Donald A.; and Snyder, Christopher A.: The Effects of Chemical Dissociation on Turbine Engine Performance with Different Fuels and Combustor Burner Temperatures. To be published.
16. Gordon, Sanford; and McBride, Bonnie J.: Computer Program for Calculation of Complex Chemical Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman-Jouguet Detonations. NASA SP-273, 1976.
17. Corban, Robert R.: Interactive-Graphic Flowpath Plotting for Turbine Engines. NASA TM-82756, 1981.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE August 1991	3. REPORT TYPE AND DATES COVERED Technical Memorandum		
4. TITLE AND SUBTITLE The Navy/NASA Engine Program (NNEP89) — A User's Manual		5. FUNDING NUMBERS  WU-505-69-50		
6. AUTHOR(S) Robert M. Plencner and Christopher A. Snyder				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Lewis Research Center Cleveland, Ohio 44135-3191		8. PERFORMING ORGANIZATION REPORT NUMBER  E-6484		
9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, D.C. 20546-0001		10. SPONSORING/MONITORING AGENCY REPORT NUMBER  NASA TM-105186		
11. SUPPLEMENTARY NOTES Responsible person, Robert M. Plencner, (216) 433-7029.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT  Unclassified - Unlimited Subject Category 07		12b. DISTRIBUTION CODE		
13. ABSTRACT (Maximum 200 words)  An engine simulation computer code called NNEP89 has been written to perform one dimensional steady state thermodynamic analysis of turbine engine cycles. By using a very flexible method of input, a set of standard components are connected at execution time to simulate almost any turbine engine configuration that the user could contemplate. The code has been used to simulate a wide range of engine cycles from turboshafts and turboprops to air-turbo-rockets and supersonic cruise variable cycle engines. Off-design performance is calculated through the use of component performance maps. A chemical equilibrium model is incorporated to adequately predict chemical dissociation as well as model virtually any fuel. NNEP89 is written in standard FORTRAN77 with clear structured programming and extensive internal documentation. The standard FORTRAN77 programming allows it to be installed onto most mainframe computers and workstations without modification. The NNEP89 code has been derived from the Navy/NASA Engine program (NNEP). NNEP89 provides many improvements and enhancements to the original NNEP code and incorporates features which make it easier to use for the novice user. NNEP89 has been written to execute the old NNEP input files without changes to the program or the input files. <del>This report serves as a comprehensive user's guide for the NNEP89 code. It incorporates a general description of the code, comprehensive input description, program flow charts and sample input and output cases.</del>				
14. SUBJECT TERMS Computer code; Engine performance; Cycle analysis; Chemical dissociation; Chemical equilibrium; Turbine engine		15. NUMBER OF PAGES 126		
		16. PRICE CODE A07		
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT	